

COLD FUSION Developer's Journal

ColdFusionJournal.com

December 2005 Volume:7 Issue:12

Data Encryption in ColdFusion 8

Misconceptions and Myths About ColdFusion ... 14

The Real Estate Sample Application PART 1 18

Toward a New Orthodoxy 30

BrowserHawk 9 by cyScape 36

Event Gateways 40

Building a Drag-and-Drop Shopping Cart with AJAX

26

Presorted
Standard
US Postage
PAID
St. Croix Press

PLUS...

Letters to the Editor

7

tag.cfc 0.1: Write your own code generator

12

ColdFusion User Groups

34



One little box. A whole lot of power.

Put it to work for you.
The shortest distance between you
and powerful web applications.

Full speed ahead. The release of Macromedia ColdFusion MX 7 is changing the whole game. This groundbreaking release introduces new features to help address your daily development challenges. These features include:



› **Structured business reporting? Check. Printable web content? Check.**

ColdFusion MX 7 provides a structured business reporting solution that will have you creating detailed business reports for anyone who needs them. You can also dynamically transform any web content into high-quality, printable documents with ease. The days of needing a third-party application to generate dynamic reports are going, going, gone.



› **Make rapid J2EE development a reality.**

So, you're heavily invested in J2EE but would love to complete projects in less time? ColdFusion MX 7 is your answer: It delivers RAD productivity on top of the power and scalability of J2EE and deploys onto standard J2EE server environments.



› **New mobile applications are a go.**

Innovative new features enable ColdFusion MX 7 applications to reach beyond the web. So you can rapidly create new applications, or extend existing ones, to connect with mobile phones and IM clients using a variety of protocols. The new world of mobile communications is exciting, and this your invitation to the party.

To learn more or take ColdFusion MX 7 for a test drive, go to:
macromedia.com/go/cfmx7_demo



Xmas Mega Pack

**SO MANY FEATURES
WE HAD TO BRING 2 SANTAS**



459 Server side Features for Dreamweaver 8

Build any dynamic website you might think of with Xmas Mega Pack 2005. This special bundle includes 21 Dreamweaver extensions that help you easily develop and maintain dynamic applications saving hundreds of hours of work. In a nutshell - all your Christmas wishes are fulfilled. Xmas Mega Pack 2005 costs \$899 and this offer is available till December 31st. Visit www.interaktonline.com/xmas/ for detailed information.



MX File Upload

Upload files and images, create thumbnails



MX RSS Reader-Writer

Use RSS feeds in your sites



MX Query Builder

Visual SQL query builder for Dreamweaver



MX User Login

Professional user login for your sites



MX Coder Pack

Code completion for Dreamweaver's programmers



MX Looper

Nested Repeats and Horizontal Loopers in Dreamweaver



MX Form Validation

Form Validation made easy



MX CSS Dynamic Menus

Create CSS pull down menus from Dreamweaver Recordsets



MX Includes

Functional Server-Side Includes in Dreamweaver



MX Send E-mail

Send e-mails from web forms



MX Dynamic Charts

Create pie, bar and line charts from Dreamweaver Recordsets



NeXTensio

Dreamweaver Extension for dynamic lists and forms



MX Kart

Shopping cart extension for Dreamweaver



MX Site Search

Search database sites



MX Widgets

Form controls for validation and enhanced usability



KTML

Online Visual HTML editor



MX CSV Import-Export

Easy CSV Import Export



XML Import-Export

Easy Data transfer with XML



MX Shop

Customizable e-commerce web application for Dreamweaver



MX Calendar

Build web calendars from Dreamweaver



MX Table Sorter

Sort your dynamic tables

work smart

www.interaktonline.com/xmas

Interakt

Complete source code and asset management in Dreamweaver MX—now possible with Surround SCM.

Dreamweaver users know a beautiful Web-based product is only skin deep. Underneath, it's a tangle of hundreds or thousands of ever changing source files. Without a good development process and strong tools, bad things happen. Surround SCM can help.

Surround SCM lets you...

Track multiple versions of your source files and easily compare and merge source code changes.

Check out files for exclusive use or work in private workspaces when collaborating on a team.

Automatically notify team members of changes to source files—push changes through your organization.

View complete audit trails of which files changed, why, and by whom.

Associate source code changes with feature requests, defects or change requests (requires additional purchase of TestTrack Pro).

Remotely access your source code repository from Dreamweaver MX.

Surround SCM adds flexible source code and digital asset control, powerful version control, and secure remote file access to Dreamweaver MX. Whether you are a team of one or one hundred, Surround SCM makes it easier to manage your source files, letting you focus on creating beautiful Web-based products.

Features:

Complete source code and digital asset control with private workspaces, automatic merging, role-based security and more.

IDE integration with Dreamweaver MX, JBuilder, Visual Studio, and other leading Web development tools.

Fast and secure remote access to your source files—work from anywhere.

Advanced branching and email notifications put you in complete control of your process.

External application triggers let you integrate Surround SCM into your Web site and product development processes.

Support for comprehensive issue management with TestTrack Pro—link changes to change requests, bug reports, feature requests and more.

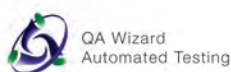
Scalable and reliable cross-platform, client/server solution supports Windows, Linux, Solaris, and Mac OS X.

Want to learn more?

Visit www.seapine.com/webdev to discover how Surround SCM can save you time and headaches. Be sure to download our white paper, **Change Management and Dreamweaver**, and learn why change management is the new must-have tool for web development.



www.seapine.com/webdev
1-888-683-6456



macromedia
ALLIANCE PARTNER



editorial advisory board

Jeremy Allaire, *founder emeritus, macromedia, inc.*
Charlie Arehart, *CTO, new atlanta communications*
Michael Dinowitz, *house of fusion, fusion authority*
Steve Drucker, *CEO, fig leaf software*
Ben Forta, *products, macromedia*
Hal Helms, *training, team macromedia*
Kevin Lynch, *chief software architect, macromedia*
Karl Moss, *principal software developer, macromedia*
Michael Smith, *president, teratech*
Bruce Van Horn, *president, netsite dynamics, LLC*

editorial**editor-in-chief**

Simon Horwith simon@sys-con.com

technical editor

Raymond Camden raymond@sys-con.com

editor

Nancy Valentine nancy@sys-con.com

associate editor

Seta Papazian seta@sys-con.com

research editor

Bahadır Karuv, PhD bahadir@sys-con.com

production**production consultant**

Jim Morgan jim@sys-con.com

lead designer

Abraham Addo abraham@sys-con.com

art director

Alex Botero alex@sys-con.com

associate art directors

Louis F. Cuffari louis@sys-con.com

Tami Beatty tami@sys-con.com

Andrea Boden andrea@sys-con.com

contributors to this issue

Laura Arguello, Steve Bryant, Joe Danziger, Nahuel Foronda,
Ben Forta, Hal Helms, Simon Horwith, Jeff Houser,
Sarge Sargent, Nicholas Tunney

editorial offices**SYS-CON MEDIA**

135 Chestnut Ridge Rd., Montvale, NJ 07645

Telephone: 201 802-3000 Fax: 201 782-9638

COLDFUSION DEVELOPER'S JOURNAL (ISSN #1523-9101)

is published monthly (12 times a year)

by SYS-CON Publications, Inc.

postmaster: send address changes to:

COLDFUSION DEVELOPER'S JOURNAL

SYS-CON MEDIA

135 Chestnut Ridge Rd., Montvale, NJ 07645

©copyright

Copyright © 2005 by SYS-CON Publications, Inc.

All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy or any information, storage and retrieval system, without written permission.

Worldwide Newsstand Distribution

Curtis Circulation Company, New Milford, NJ

FOR LIST RENTAL INFORMATION:

Kevin Collopy: 845 731-2684, kevin.collopy@edithroman.com

Frank Cipolla: 845 731-3832, frank.cipolla@epostdirect.com

For promotional reprints, contact reprint coordinator Megan Mussa, megan@sys-con.com. SYS-CON Publications, Inc., reserves the right to revise, republish and authorize its readers to use the articles submitted for publication.

All brand and product names used on these pages are trade names, service marks, or trademarks of their respective companies.

2005 – The Year in Review



By Simon Horwith

2005 has been one of the most significant years

in history for ColdFusion developers and Web developers in general.

2006 looks to be a very promising year as well, between the impending release of Flex 2, the beginning of seeing what Adobe does with the product, and the continued development of ColdFusion 8, code named "Scorpio." Still, at the end of the year it's good to look back on the legacy we will inherit and build-on in the year(s) to come.

This past year saw the release of Dreamweaver 8. This release of Dreamweaver was definitely the best to date – the product appears to be more stable than ever: there is better support for working with ColdFusion Components, better XML support, and excellent major enhancements to its CSS support. In 2005 we also saw a huge acceptance of the CFclipse project. Many ColdFusion developers are now using this free IDE to meet all of their development needs, and (especially for those who don't like using Dreamweaver to write their code) this has given us back what can only be referred to as a "real developers" IDE.

The community celebrated and participated in a couple of major events this past year as well. Developers converged on Newton and congregated at their local user groups in order to celebrate ColdFusion's 10th birthday. The annual CFun conference was rebranded CFUnited and saw a major increase in attendance and sponsorship, and has now become the largest and most significant annual ColdFusion event. The Fusebox conference was also reformatted to include all frameworks, which made for a terrific event.

Speaking of frameworks, the emergence

of several new frameworks and new versions of existing frameworks is a noteworthy trend in 2005. Frameworks in general have had much more widespread use and acceptance. More developers are beginning to publicly release implementations of design patterns and modules to make common tasks easier. There is more discussion on blogs, list servers, and conferences about object-oriented CF. These things are all symptoms of a trend: more developers are beginning to approach ColdFusion development as serious software development.

This past year saw the release of the eagerly anticipated ColdFusion MX 7, and it was well worth the wait. CFMX 7 was, in fact, the most significant feature release I can remember since the first version of ColdFusion was released over 10 years ago. A new application framework, PDF and FlashPaper support, Flash Forms, new reporting formats and features, a ColdFusion Report IDE, many new packaging and deployment options including support for .EAR/.WAR packaging, a new version of Verity with support for many new search and indexing options, and the new Event Gateway framework to allow ColdFusion to do so much more than simply respond to HTTP requests are among the most significant features. I know the team is looking to make Scorpio the best release ever, but they've sure got their work cut out for them.

– CONTINUED ON PAGE 39

About the Author

Simon Horwith is the editor-in-chief of ColdFusion Developer's Journal. Simon is a Macromedia Certified Master Instructor and a member of Team Macromedia. He has also been a contributing author of several books and technical papers. You can read his blog at www.horwith.com

simon@horwith.com



Dedicated Server Packages Starting at \$189/mo.

All dedicated servers include:

- ▶ FREE STATS SOFTWARE!
- ▶ No long term commitments!
- ▶ FREE SQL server access!
- ▶ FREE MAIL SOFTWARE!
- ▶ Fair and simple pricing!
- ▶ Optional server maintenance!

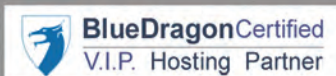
As one of the premier ColdFusion hosting community leaders, CFDynamics is constantly looking for ways to provide **better service** to ensure your satisfaction. Not only do we offer the **finest in shared hosting plans**, but we now offer the **finest in 100% dedicated server plans**! Now you can afford the freedom of having your own dedicated server!

When your needs have outgrown shared hosting look to CFDynamics for **total freedom**. With dedicated server packages they're not offering an oxymoron; "virtually private" or "virtually dedicated" is **NEITHER** private nor dedicated. CFDynamics offers a solution that is **100% completely dedicated**. They don't play games with the fake stuff; CFDynamics only offers the real deal. Real Service. Real Satisfaction. Real Value.

Real Freedom.



Paper|Thin Partner



Visit us online or call to order!

president & ceo

Fuat Kircaali fuat@sys-con.com

vp, business development

Grisha Davida grisha@sys-con.com

group publisher

Jeremy Geelan jeremy@sys-con.com

advertising**senior vp, sales & marketing**

Carmen Gonzalez carmen@sys-con.com

vp, sales & marketing

Miles Silverman miles@sys-con.com

advertising director

Robyn Forma robyn@sys-con.com

advertising manager

Megan Mussa megan@sys-con.com

sales & marketing director

Dennis Leavey dennis@sys-con.com

associate sales manager

Kerry Mealia kerry@sys-con.com

sys-con events**president, events**

Grisha Davida grisha@sys-con.com

national sales manager

Jim Hanchrow jimh@sys-con.com

customer relations**circulation service coordinator**

Edna Earle Russell edna@sys-con.com

manager, idj store

Brunilda Staropoli bruni@sys-con.com

sys-con.com**vp, information systems**

Robert Diamond robert@sys-con.com

web designer

Stephen Kilmurray stephen@sys-con.com

online editor

Roger Strukhoff roger@sys-con.com

accounting**financial analyst**

Joan LaRose joan@sys-con.com

accounts payable

Betty White betty@sys-con.com

accounts receivable

Gail Naples gailn@sys-con.com

subscriptions**Subscribe@sys-con.com****Call 1-888-303-5282**

For subscriptions and requests for bulk orders,
please send your letters to Subscription Department

Cover Price: \$8.99/issue

Domestic: \$89.99/yr (12 issues)

Canada/Mexico: \$99.99/yr

All other countries \$129.99/yr

(U.S. Banks or Money Orders)

Back issues: \$12 U.S. \$15 all others

Letters to the Editor

Is JavaScript the Essence of AJAX?

["What Is AJAX?" Part 1 by Rob Gonda, Vol. 7, issue 9]

Some people will claim that they can provide you with AJAX functionality without writing a single line of JavaScript; I disagree.

Surely JavaScript is the whole "essence" of AJAX?

—JSrocks

It is indeed. AJAX involves combining any server scripting language with client side, hence, JavaScript. I can see how you could probably automate two related select boxes without writing JavaScript, but certainly not an RIA or Web 2.0 application.

—Rob Gonda

It's been possible to use the concept behind AJAX since browsers introduced the XMLHttpRequest function in 1999.

But what was then the disconnect that prevented developers from making use of it until Jesse James Garrett kick-started the whole AJAX boom by giving it a "name"?

—queZZtion

Perhaps, nothing really prevented anyone from using it; Jesse James Garrett just had to come up with the idea/concept. By the way, don't miss Jesse at the Ajax Seminar (www.ajaxseminar.com)

—Rob Gonda

Developing AJAX Applications

["AJAX: Making the HTML User Experience Almost As Pleasant as Flash: Part 2" by Rob Gonda, Vol. 7, issue 10]

As AJAX gives Web applications a fresh look, the rising cost of developing AJAX applications is challenging its way to success.

That's the reason we founded the ZK project: to make AJAX transparent to app developers.

—Jim

First of all, I really disagree on the rising cost of developing AJAX applications. Au contraire, the cost is going down as more and more developers get into it.

Although I usually don't check seeding or spam, I decided to check the ZK project.

It definitely helps with dhtml and look and feel, but it does not automate AJAX at all; it makes it actually more difficult. It seems that in order to make it work, you need to add your Java server-side code within your view. That will make maintenance really tedious.

—Rob Gonda

"First of all, I really disagree on rising cost of developing AJAX applications."

Adding an AJAX feature means an extra task, no matter how simple it is. It gets worse when we add more code to the client. It means you might have to replicate business logic to clients. It also means you have to maintain two copies of codes.

"It definitely helps with dhtml and look and feel, but it does not automate AJAX at all, it makes it actually more difficult. It seems like that in order to make it work you need to add your Java server-side code within your view. That will make maintenance really tedious."

Whether to embed codes in the view is up to developers, not the framework itself. It is designed to speed up prototyping and customization. If MVC or other design patterns are required, developers need only to provide a map between components to the real class they want.

On the other hand, traditional AJAX apps required developers to embed JavaScript into HTML pages.

AJAX has different meanings to different people. For us, it means a technology to enable a rich user interface that communicates with the back-end server. What ZK does is to abstract the interaction and communication to Java level and make to the server side, not JavaScript or decorated HTML, not at the client that you are used to.

—Jim

...

Data Encryption in ColdFusion

An overview of the built-in features



By Jeff Houser

It is likely that at some point in your development career you had to deal with sensitive data. It might have been credit card numbers in an e-commerce site, or an employee identification number on an intranet. Perhaps you were

setting up a security scheme and wanted to protect the passwords of the user.

Maybe you wanted to encrypt some user data before sending it off to a business partner via a Web service, or setting a cookie on the user's machine? For whatever the reason you probably took a look at ColdFusion's built-in functions for encrypting data.

Unfortunately, in versions of ColdFusion prior to the seventh release, ColdFusion didn't offer much in the way of built-in data encryption. You had to go looking outside of the built-in functionality, and often purchase a custom tag to do anything advanced with encryption. Thankfully, ColdFusion MX7 expanded the built-in encryption set and gives us a much broader range of features without going outside the language. This article will tell you about those features and offer some suggestions on where to go when you need more.

Password Encryption

Have you ever come across a Website that forces you to register to see their content? If so, have you ever tried to register only to be told that you already have an account? In the dark reaches of your mind, you have a vague memory of trying to access some content on their site many years ago. They won't let you re-register because you already have an account, so you click that "I forgot my password" box. Many sites will not reveal your password to you. A common approach is for you to provide an e-mail address, and then

the site will reset your e-mail address and e-mail you the new one. Another approach might be for you to answer a series of password questions, after which you will be able to reset your password.

Why do you have to reset the password, instead of the site revealing the previous password to you? It's because they don't know the password. As part of the security process, they encrypt all passwords in their database, using a one-way algorithm. A one-way algorithm is a method of encryption where the encrypted string cannot be decrypted. There is no way to get the original data from the decrypted string. In ColdFusion you can set up this type of encryption using the hash function.

The hash function has three arguments:

- **String:** The string argument specifies the data you want to encrypt.
- **Algorithm:** The algorithm argument specifies the algorithm you want to use to generate your encrypted string. The algorithm selected will affect the length of the encrypted string. CFMX_COMPAT or MD5 will generate a 32-character string, compatible with previous version of CFMX. SHA will generate a 28-character string, SHA-256 a 44-character string, SHA-384 a 64-character string, and SHA-512 a 88 character string.
- **Encoding:** Encoding is an option attribute that is used to specify the encoding to use when converting the string to byte data. If you leave out this attribute, the defaultCharset value, set in neo-runtime.xml, is used.

So, how can you apply this to your own development? Suppose you had a registration form on your Website. The users fill out a bunch of information, including username and password. The form probably looks something this:

```
<form action="save.cfm" method="post">
    username: <input type="text" name="username">
    password: <input type="password" name="password">
    .. other user stuff here
</form>
```



The user steps through the registration process and at some point they submit the form. Your verification code decides that there is no problem with the user's submission, and you'll run an insert query like this:

Complex JAVA J2EE Hosting made easy.

WebAppCabaretsm

<http://www.webappcabaret.com/jdj.jsp>
1.866.256.7973



JAVA J2EE-Ready Managed Dedicated Hosting Plans:

Xeon I

**SAMEDAY
SETUP**

Dual 2.8 GHz Xeons
2GB RAM
Dual 73GB SCSI
1U Server
Firewall
Linux
Monitoring
NGASI Manager

\$279
monthly

Pentium 4 I

**SAMEDAY
SETUP**

**FREE
SETUP**

2.4 GHz P4
2GB RAM
Dual 80GB ATA
1U Server
Firewall
Linux
Monitoring
NGASI Manager

\$199
monthly
2nd month
FREE

4Balance I

1 Database Server
and 2 Application
Servers connected
to 1
dedicated load
balancing device.
Dual Xeons.
High-Availability.

\$1724
monthly

At **WebAppCabaret** we specialize in **JAVA J2EE Hosting**, featuring **managed dedicated servers** preloaded with most open source JAVA technologies.

PRELOADED WITH:

JDK1.4 . JDK1.5 . Tomcat . JBoss . Struts . ANT . Spring . Hibernate
Apache . MySQL . PostgreSQL . Portals . CRM . CMS . Blogs . Frameworks
All easily manage via a web based control panel.

Details:

- All Servers installed with the latest Enterprise Linux
- Firewall Protection
- Up to 60 GB daily on site backup included at no extra charge per server.
- Database on site backup every 2 hours
- Daily off site database backup
- A spare server is always available in case one of the server goes down
- Intrusion detection.
- 24x7 Server and application monitoring with automatic self healing
- The Latest Bug fixes and Security updates.
- Tier 1 Data Center. 100% Network Uptime Guarantee
- Guaranteed Reliability backed by industry-leading Service Level Agreements

Log on now at <http://www.webappcabaret.com/jdj.jsp> or call today at **1.866.256.7973**



WebAppCabaretsm

JAVA J2EE Hosting

Prices, plans, and terms subject to change without notice. Please log on to our website for the latest price and terms. Copyright © 1999-2005 WebAppShowcase • All rights reserved • Various trademarks held by their respective owners.

```
<cfquery name="savestring" datasource="Mydb">
  insert into Users(username, password, otheruserstuff)
  values ('#form.username#', '#hash(form.password)#', 'otheruserstuff')
</cfquery>
```

Although you collected a plain text password, you don't actually store it in that format. The information is hashed when it goes into the database. How do you actually compare that information when the user tries to log back in? Easy, you hash the password they entered and compare the hashed results with the hashed value in the database.

Suppose this is your login form:

```
<form action="login.cfm" method="post">
  username: <input type="text" name="username">
  password: <input type="password" name="password">
</form>
```

(Hey, that code snippet looks familiar.) The user clicks submit and, on the login page, you can verify the login like this:

```
<cfquery name="Login" datasource="Mydb">
  select * Users
  where username = '#form.username#' and
        password = '#hash(form.password)#'
</cfquery>
```

There you have it. More information about the hash function can be found in the livedocs <http://livedocs.macromedia.com/coldfusion/7/htmldocs/00000503.htm#1105551>.

Data Decryption

One drawback of the hash function is that you are never able to take the encrypted value and get the original string. Sometimes, you'll need to see the data behind your encrypted strings. ColdFusion has two functions to help us with this, aptly named encrypt and decrypt. Both functions have a similar set of arguments:

- **String:** The string argument is required. It is either the string that you want to encrypt (for the encrypt function) or the encrypted string you want to decrypt (for the decrypt function)
- **Key:** The key argument is used to specify the key for encryption and decryption. You use this value to go from the

original value to an encrypted string, and to go from the encrypted string back to the original value. If you are using CFMX_COMPAT (which is included for backward compatibility), any string can be used in the encrypt function. Otherwise, you should use the GenerateSecretKey function to create a key for the particular algorithm. For decryption, you should use the same key value that you used to encrypt the value in the first place. GenerateSecretKey accepts one argument, which is the name of the algorithm.

- **Algorithm:** The algorithm argument accepts the name of the encryption library that you want to use. CFMX_COMPAT is the default value, making this argument optional. Other values are AES, Blowfish, DES, or DESEDE.
- **Encoding:** The encoding argument is used to specify the binary encoding that is used to represent the data as a string. UU is the default value, with the alternate's being Base64 or Hex.
- **IVorSalt:** If you are using a block encryption algorithm, you need to specify the initialization vector (IV) for compatibility with other systems. You put that value here. For password-based encryption, specify the salt value here. A salt value is random data that is part of the session key and helps deter brute force decryption attempts.
- **Iterations:** The iterations argument specifies the number of iterations that are taken to transform the password into a binary key. This argument is not used for block encryption algorithms.

For this example, I'll assume you need to integrate two systems and must send data from one to the other, via an HTTP post. (Believe it or not, my experience is that this method is still in wider use than SOAP Web services.) First, you need to create the page to send the data:

```
<cfset data = encrypt('Jeff','12345')>
<cfhttp url="http://localhost//recieve.cfm"
  method="post" result="MyResults" >
  <cfhttpparam name="data" value="#data#"
  type="formfield">
</cfhttp>
<cfdump var="#variables#">
```

I named this file "send.cfm". It en-

crypts a piece of data and uses cfhttp to submit it onto a receive.cfm page in the root directory of your local Web server. The variable's scope is cfdumped. This will show us our encrypted value and the results of the HTTP post (which will include the decrypted value). Take a look at receive.cfm:

```
<cfset decryptedstring = decrypt(form.
data,'12345')>
```

```
<cfoutput>
  #decryptedstring#
</cfoutput>
```

This page is simpler than the previous. It decrypts the data from the form post, then outputs it. In the real world, you'd use a key that was a little more complicated than "12345" (most likely generating something with the GenerateSecretKey function), and your data will probably not be hard coded, but something taken out of a database. When the time comes, you can expand on the concepts in this example to make something a little more complex. You can read the livedoc information on encrypt at <http://livedocs.macromedia.com/coldfusion/7/htmldocs/00000457.htm#1104201>, decrypt at <http://livedocs.macromedia.com/coldfusion/7/htmldocs/00000452.htm#1103962>, and GenerateSecretKey at <http://livedocs.macromedia.com/coldfusion/7/htmldocs/00000468.htm#4992278>.

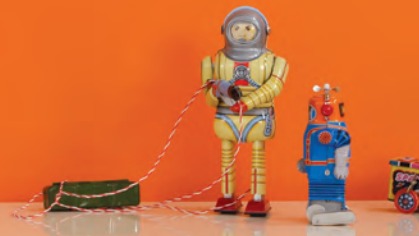
Where to Go From Here

What's next? One thing that ColdFusion does not (yet) natively support is a public/private key encryption scheme. In this type of scheme, you would use one key to encrypt data and another to decrypt the data. If you were sending data to me, you would encrypt it with my public key, then it can only be decrypted with my private key and you know that I'll be the only one reading it. Or perhaps I wanted to verify that you were truly the one sending me the data. If you encrypt it using your private key, I can decrypt using your public key. Two keys increase the amount of encryption available.

— CONTINUED ON PAGE 17



WEB DEV GURU seeks an Integrated Software Suite that speaks my language (XHTML) and won't cramp my style. Must play well with XML, CSS and others. I'm not superficial, but I like a nice code view. Clutter isn't cute.



Different people. Different needs. One suite solution.

With the latest versions of Macromedia Dreamweaver®, Flash® Professional, Fireworks®, Contribute™, and FlashPaper™, the new Studio 8 is quite a catch. To meet Studio 8 and find all the web design and development tools you need, visit www.macromedia.com/go/8_studio8.

macromedia®
STUDIO 8

tag.cfc 0.1

Write your own code generator



By Steve Bryant

Back when I had some free time, I started working on my own code generator, partly because other code generators create code slightly different from my preferences and partly for the challenge.

I have since gotten too busy to finish it, but I went ahead and finished the kernel component of the system – the one that actually generates the code. To download the component go to www.bryantwebconsulting.com/cfcs.

One advantage of ColdFusion being a tag-based language is that generating code for ColdFusion largely means writing tags.

The best way that I can think to show how tag.cfc works is by example. cf_sebForm.cfc and cf_sebField.cfc are both components that are inherited from tag.cfc. They are each used to generate the code for my cf_sebForm and cf_sebField custom tags.

These components are actually very simple. They extend tag.cfc and each have two methods: vtml() and schema(), which return the VTML for the tag and the XML Schema for the tag, respectively.

Here is some example code using these components to output the code needed for these custom tags:

```
<cfset sebForm = CreateObject("component","cf_sebForm").init()>
<cfset fields = ArrayNew(1)>

<cfset sebForm.setAttribute("formname","myform")>
<cfset sebForm.setAttribute("librarypath","/lib/")>

<cfset fields[1] = CreateObject("component","cf_sebField").init()>
<cfset fields[1].setAttribute("fieldname","blah")>
<cfset fields[1].setAttribute("label","Hello")>
<cfset fields[1].setAttribute("type","text")>
<cfset fields[2] = CreateObject("component","cf_sebField").init()>
<cfset fields[2].setAttribute("label","Submit")>
<cfset fields[2].setAttribute("type","submit")>
<cfset sebForm.addTag(fields[1])>
<cfset sebForm.addTag(fields[2])>
<cfoutput>
<pre>#HTMLEditFormat(sebForm.write())#</pre>
</cfoutput>
```

The code that this code would output is the following:


```
<cf_sebForm formname="myform" librarypath="/lib/">
  <cf_sebField fieldname="blah" type="text" label="Hello" />
  <cf_sebField type="submit" label="Submit" />
</cf_sebForm>
```

In order for the previous code to be useful, of course, you wouldn't want the values to be hard-coded. You would need to get those values from a form or from data in a database, for example. While tag.cfc doesn't handle that part for you, it will handle most of the last step of code generation once you have gathered that information.

Incidentally, when I was working on my own code generator (a task I hope to return to someday), I used my DataMgr component to get the structure of the database. Since it works the same across multiple databases, it makes for a nice cross-database solution.

You can download tag.cfc from my site. I also have two sets of tag CFCs that inherit from tag.cfc. The first, "CFCs", includes components to generate cfcomponent, cffunction and cfargument. The second, "sebtags", is a set of CFCs used to generate my custom tags.

This code illustrates my best understanding of good OO code, so it should be a good example of such. That being said, if someone can show why it isn't a good example of such, let me know.

Good luck! 

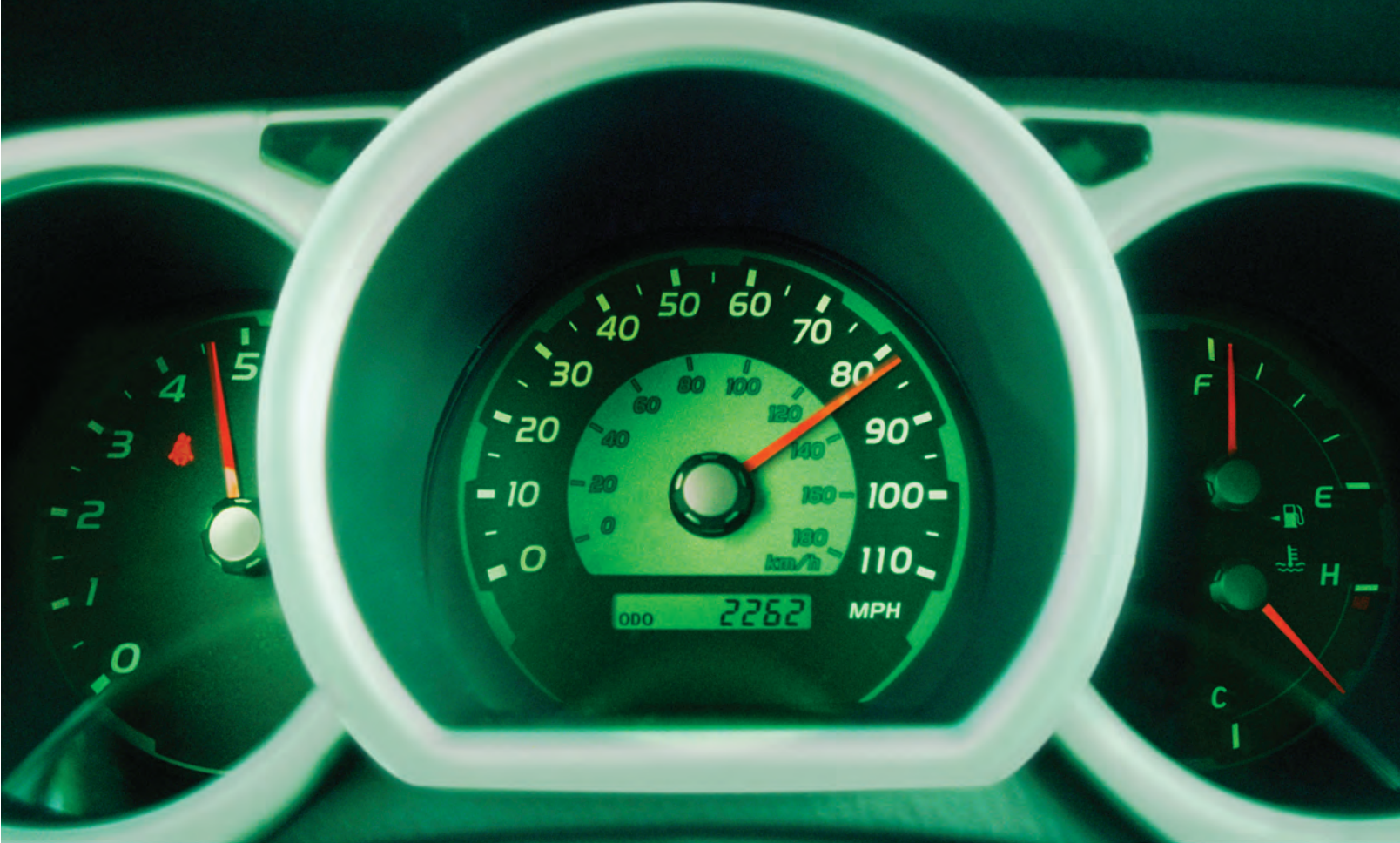


This article was reprinted with permission from Steve Bryant's blog: <http://steve.coldfusionjournal.com>

About the Author

Steve Bryant is the owner of Bryant Web Consulting LLC (www.bryantwebconsulting.com) and teaches ColdFusion at Breakaway Interactive (www.breakawayinteractive.com). He received his BA in philosophy at Oklahoma State University and still has no idea how that led to a career in Web development.

steve@bryantwebconsulting.com



Rev Up Your Flash Video

Stay Ahead of the Competition With VitalStream and the Enhanced Video Features in Flash 8

With over two years of experience in delivering much of today's most popular media, VitalStream® is the first and most experienced Flash™ video streaming service provider.

Enhanced Flash 8 Video Features:

- New VP6 codec delivers higher quality video at the same bit rate
- 8-bit alpha channel transparency enables you to blend video with other elements
- Improved live video capabilities

VitalStream Complete Toolset for Flash:

- MediaConsole®
- MediaOps™ SDK
- Flash Authentication
- Reporting Dashboard



*Integrate Streaming Media
Into Your Flash Projects*

Take Advantage of the Enhanced Video Features in Macromedia Flash 8
Call (800) 254-7554 or Download Tutorials at www.vitalstream.com/go/mxdj

Misconceptions and Myths About ColdFusion . . .

. . . debunked



By Simon Horwith

As someone who stands in front of audiences and evangelizes ColdFusion, often times I not only find myself preaching to the choir, but I sometimes find myself defending the product to misinformed or

disheartened individuals. In the role of developer I, and many of our readers, find themselves in the same predicament.

In fact, the real battle to dispel myths and misconceptions about ColdFusion is not taking place at conferences and user group meetings. It's happening in the workplace, and every ColdFusion developer shares the responsibility to dispel any misconceptions or myths they encounter and to defend the use of their beloved product. Fortunately, one thing that's certainly true about ColdFusion developers is that they are enthusiastic, even fanatic, about the product; so there's no shortage of people to defend its use. Unfortunately, we don't all know what to say or how to respond to every objection thrown our way. That is what this article is about – to justify the use of ColdFusion or even its existence within an organization.

Each of the myths and misconceptions surrounding ColdFusion could be classified into one of three categories:

1. Misconceptions about its capabilities
2. Misconceptions about its community or corporate support
3. Misconceptions about its cost

The majority of the misconceptions about ColdFusion surround its capabilities. We've all been there; having to defend against the notion that ColdFusion is a

toy; that it's not capable of delivering enterprise solutions; that CFML is not a "real" programming language. All of these ideas derive from a lack of understanding about what ColdFusion is, not about what ColdFusion is not. Let's start from the beginning: ColdFusion is a J2EE application. What that means is that the ColdFusion server is a Java application designed to run on top of the J2EE platform (a J2EE application server). ColdFusion is a Java application that parses CFML, its proprietary tag-based language, in order to generate and run Java code under the hood. CFML, the tag-based language that ColdFusion developers use to write applications, helps to make common development tasks trivial. This, more than anything, is what makes CFML the most rapid development environment on the Web. What all of this ultimately means is that ColdFusion is the most rapid environment for developing Java applications.

The idea that ColdFusion isn't suited for enterprise applications is absurd – unless you believe Java isn't suited for enterprise application development. ColdFusion supports all of the functionality in Java (you even use Java to extend the server via the ColdFusion Event Gateway framework) and scales as well as any Java application. In fact, you could argue that it's better suited than "plain" Java as an enterprise application platform because CF offers faster development and debugging times (and easier deployment), which can be a major factor in the success of large-scale enterprise applications.

Another common misconception about the capabilities of ColdFusion is that it isn't secure. There's no foundation for a statement like this – the server itself is very secure, and applications written for ColdFusion are as secure as the author chooses to make them. Sure, it's possible to create CF applications that aren't secure just like it's possible to create vulnerable .NET, PHP, PERL, RUBY, or Java applications as well. This is a question of the competency of the developer, not the platform. Speaking of which, I'll lay to rest the misconception that bothers me most: that "real developers" don't use ColdFusion.



Are there poorly written ColdFusion applications? Sure. Are all ColdFusion applications poorly written? No – far from it, in fact. The truth is that because ColdFusion is somewhat easier to learn than “traditional” programming languages (largely due to its tag syntax), many CF developers don’t have computer science backgrounds. There are plenty of us who write extremely advanced code and specialize in developing complex applications. Of course, a vast majority of the applications on the Web really don’t call for this level of complexity, so the number of developers who specialize in developing very advanced applications is small in proportion to the total number of people using ColdFusion. The fact that ColdFusion is so easy that anyone can learn it is a testament to the language, not a drawback. I’ve worked on many ColdFusion applications alongside Java developers who were skeptics at first but who, after spending a few hours talking with me and a day or two writing CF code, became converts and avid fans of CF. Where is it said that because something is easy, it’s no good? Isn’t that the point of IDEs (and computers for that matter) – to make life easier? Show me a developer who wants their job to be more difficult and take longer than need be, and I’ll show you a fool. ColdFusion developers are no fools.

It’s also worth noting that there is a much larger number of well-written large-scale enterprise applications written in ColdFusion than most people realize; this is because a huge percentage of the organizations using CF are doing so behind corporate firewalls and on secure networks for intranet and other internal applications. In over 10 years of ColdFusion development for both private and government entities in and around Washington, D.C. (somewhat of a mecca of CF development), I can state with authority that there’s a lot more ColdFusion out there than people realize. Many government and large corporate sites, behind their public-facing static HTML façades, are relying on mission-critical CF applications to get things done on a daily basis.

Another misconception is about the price of ColdFusion – some people don’t like the fact that you have to pay for CF. Whenever I hear this argument from somebody, I always tell them that the best things in life aren’t free. No, this rebuttal hasn’t worked for me yet, but it gets the occasional laugh. In reality, what needs to be understood is that nothing is free. People will often question why .NET or PHP or PERL don’t cost anything but ColdFusion does. There are two things that people with this misconception need to understand.

First, ColdFusion has a huge number of features out of the box. If you tried to create a .NET environment with all the same features (Verity searching, PDF generation, Flash Remoting, etc.) the cost would be much higher than the cost of a single ColdFusion license. Note that I said “tried to create...” – good luck getting .NET to run on all of the same operating systems that ColdFusion supports, or getting it to natively support Flash form generation the way that ColdFusion does.

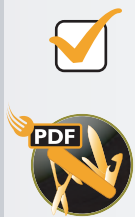
The second thing that people with pricing concerns need to understand is the immense savings in development time you get when using ColdFusion. Ignoring abnormal assign-

What’s your PDF?



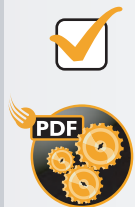
Precise Document Formatting

With activePDF Server, you gain full control over your PDF output with conversion options that allow you to specify page size, compression and resolution options, embed text, create bookmarks, concatenate to existing files, and more. Licensed per server, you can easily add PDF generation to virtually any Windows application.



Populate Dynamic Forms

With activePDF Toolkit’s form-filling capabilities, you can dynamically populate PDF forms with data and images from a database, allow users using only Adobe Reader to fill-in and save forms and use PDF forms as document templates to precisely control image placement and resizing. With Toolkit’s robust API, the automation of virtually any PDF manipulation task becomes possible - append, stamp, stitch, merge, paint, secure PDF and more.



Promote Digital Fidelity

Do you need to standardize PDF output within your enterprise? With DocConverter, you can easily use built-in support for “watched” folders to implement server-side PDF generation in a matter of minutes, with full control over the PDF output at the server level. Or, use DocConverter’s programmable COM object to integrate convert-to-PDF functionality within your enterprise application.



Present Data Fashionably

Ensuring precise layout of an HTML document can be a nightmare, especially when printing. PDF guarantees pixel-perfect layout every time as what you see is what you print. With activePDF WebGrabber, you can dynamically convert any URL, HTML stream, or HTML file to PDF on the fly, while maintaining embedded styles.

Download your
free trial version today at
www.activePDF.com

Copyright © 2004, activePDF, Inc. All Rights Reserved.
“ACTIVEPDF”, “Leading the iPaper Revolution” and the
activePDF logo are registered trademarks of activePDF,
Inc. All activePDF product names are trademarks of
activePDF, Inc.



ments, a typical ColdFusion project involves anywhere from \$25,000 to \$250,000 of billable man hours writing code. Given the rapid nature of ColdFusion development and debugging, it's a safe assumption that the savings in development costs for a \$25,000 CF project will more than make up for the cost of buying an enterprise license of ColdFusion. The bottom line is that CF is cheap – development isn't. When faced with the “we can't afford CF” argument, my attitude is generally one of “you can't afford not to use CF.” For those folks who have seriously tiny budgets, there is always the option of hosting the site on a shared server and not taking on the cost of buying a license at all.


Before addressing the last category of misconceptions about ColdFusion, along the lines of cost there is one other thing to consider – are those “free” platforms really free? Most of them are, if you ignore the extra development/debugging costs, but there are some considerations with regards to .NET. .NET, the most frequently used alternative to ColdFusion, is often regarded as free. True, the platform does not cost anything, but there are further costs. In order to develop .NET applications, every developer working on a project will require a licensed copy of Visual Studio .NET – and that is not free. If you want to do ColdFusion development with Dreamweaver, it will cost about the same as a copy of Visual Studio, but you also have the option of using Eclipse for all ColdFusion development, and Eclipse (and the CFEclipse plug-in) is available for a free download (it's also my preferred IDE as well as the preferred IDE of many other developers). In order to run .NET on a production server you will also need to buy a Windows server license (2000 Enterprise or Server, 2003 Enterprise or Server, or 2003 Web Edition) – again, this is not free by any means. If costs are a real concern, ColdFusion is supported on many operating systems, including Linux (which is free). These additional costs associated with .NET aren't usually a factor in the decision-making process when evaluating CF versus .NET, but they're worth noting.

The last type of misconception surrounds corporate support for ColdFusion. Two primary myths exist with regards to this. Some people are under the impression that once you buy ColdFusion, you are on your own with no support. Like all of the

other Web development platforms, there is massive community support for ColdFusion. In fact, in my experience, the CF community is much friendlier than any other development community that I've interacted with (and I've interacted with a lot of them over the years). What about corporate support? Macromedia/Adobe has several different support agreements available in order to meet the needs of their customers. There are also many other companies (my employer, AboutWeb LLC, is one of them) that offer support agreements for both ColdFusion and the applications running on top of it. Getting back to Macromedia, they offer support in other ways as well. One way is via the Macromedia (Adobe) Website – particularly through the forums and articles/tech notes that are searchable via their knowledge base. They also help the community help itself by way of their User Group program – through which they help local groups of developers organize meetings, presentations, and speakers. Macromedia also supports the community by way of the Team Macromedia program – individuals who are considered leaders in the community have special access to Macromedia resources (such as the support and product development teams) and are kept up to date with the latest information (and products) even before it's released. Then there's training – Macromedia Training develops great hands-on courses and materials and is very good about controlling and maintaining an excellent caliber of services available via Macromedia Authorized Training Partners and Macromedia Certified Instructors. Last, Macromedia has a good network of preferred service providers. If you call Macromedia looking for help, whether it's training, troubleshooting, or actual development that you need, and if your needs are not within the realm of what they are able to offer you, Macromedia has a list of top companies that they will refer you to and put you in contact with.

The second myth about the support for ColdFusion is that its future is uncertain. I'm not sure why this misconception exists – ColdFusion has been around for over 10 years now and continues to grow both in terms of popularity and feature strength. Though the product has changed ownership twice, continuing to support and enhance the product has been a priority of every company

that has acquired ColdFusion. Perhaps some people didn't have total confidence that Macromedia was financially strong enough to feel good about their future (though there was never any justification for such a theory), but nobody should doubt the stability and longevity of Adobe, which is even stronger now that they have acquired Macromedia. ColdFusion makes Adobe a big player in the application server/Web development platform market, and they have made it clear that they have big plans for CF. So the future looks very bright, more so than ever, for ColdFusion.


It's sad that we developers sometimes have to spend our valuable time arguing the same points over and over from one naysayer to the next. Try not to think of it as arguing; what we do is educate, not argue. Hopefully, this article will help those of you who find yourselves in situations where you have to dispel myths about CF better prepared to do so. As I said in the beginning of the article, I often end up preaching to the choir, so better yet, I hope this article finds its way into the hands of individuals who have misconceptions of their own about this terrific product. ColdFusion, like anything else, isn't the one solution to every problem... but when evaluating all of your options it's important to be well informed. More often than not, for Web applications, it is the best solution. If you find yourself or someone else in a situation where they think otherwise, point them at this article, because their reason for leaning one way or the other may well be founded in misconceptions. As a last resort, if any of you still have questions about whether or not CF is the right solution to use on a project, feel free to send me an e-mail at shorwith@aboutweb.com and I'll be sure to help set the record straight. 

About the Author

Simon Horwith is the editor-in-chief of ColdFusion Developer's Journal. Simon is a Macromedia Certified Master Instructor and a member of Team Macromedia. He has also been a contributing author of several books and technical papers. You can read his blog at www.horwith.com

simon@horwith.com

Data Encryption in ColdFusion

There are two public / private key tags listed in the Macromedia Developer's exchange, one from <http://www.tamuri.com/> and one from <http://www.digitaloutlook.com/>. Both require that you install PGP, which (I'm warning you) will prevent ColdFusion's wsconfig tool from working. It is a roadblock if you are using CFMX in the J2EE configuration. The Tamuri product only works with PGP 8.0, which appears to no longer be available, and I was never able to get it to work. I had much better luck with the Digital Outlook tag, but the tag stopped working for an unknown reason and even support couldn't help me get it back to a functional point. Hopefully Adobe will include this functionality natively in the next release. 

About the Author

Jeff Houser has been working with computers for over 20 years. He owns a DotComIt, a web consulting company, manages the CT Macromedia User Group, and routinely speaks and writes about development issues. You can find out what he's up to by checking his Blog at www.jeffryhouser.com.

jeff@instantcoldfusion.com

CFDJ Advertiser Index

ADVERTISER	URL	PHONE	PAGE
ACTIVEPDF	WWW.ACTIVEPDF.COM	866-468-6733	15
CFDYNAMICS	WWW.CFDYNAMICS.COM	866-233-9626	6
COMMUNITYMX	WWW.COMMUNITYMX.COM/TRIAL		51
HOSTMYSITE	WWW.HOSTMYSITE.COM	877-215-4678	25
INTERAKT	WWW.INTERAKTONLINE.COM/	4031-401-68-193	
INTERGRAL	WWW.FUSION-REACTOR.COM		21
INTERMEDIA	WWW.INTERMEDIA.NET	888-379-7729	52
MACROMEDIA	MACROMEDIA.COM/GO/CFMX7_DEMO	415-252-2000	2
MACROMEDIA	WWW.MACROMEDIA.COM/GO/8_STUDIO8	415-252-2000	11
PAPERTHIN	WWW.PAPERTHIN.COM	800-940-3087	31
SAVVY	WWW.BESAVVY.COM	866-870-6358	17
SEAPINE	WWW.SEAPINE.COM/WEBDEV	888-683-6456	4
SYSTEMS ALLIANCE	WWW.SITEEXECUTIVE.COM/CFDJ	877-797-2554	23
VITALSTREAM	WWW.VITALSTREAM.COM	800-254-7554	13
WEBAPPCABARET	WWW.WEBAPPCABARET.COM/JDJ.JSP	866-256-7973	9
WEBAPPER	WWW.SEEFUSION.COM		31

General Conditions: The Publisher reserves the right to refuse any advertising not meeting the standards that are set to protect the high editorial quality of. All advertising is subject to approval by the Publisher. The Publisher assumes no liability for any costs or damages incurred if for any reason the Publisher fails to publish an advertisement. In no event shall the Publisher be liable for any costs or damages in excess of the cost of the advertisement as a result of a mistake in the advertisement or for any other reason. The Advertiser is fully responsible for all financial liability and terms of the contract executed by the agents or agencies who are acting on behalf of the Advertiser. Conditions set in this document (except the rates) are subject to change by the Publisher without notice. No conditions other than those set forth in this "General Conditions Document" shall be binding upon the Publisher. Advertisers (and their agencies) are fully responsible for the content of their advertisements printed in ColdFusion Developer's Journal. Advertisements are to be printed at the discretion of the Publisher. This disclaimer includes the positioning of the advertisement, except for "preferred positions" described in the rate table. Cancellations and changes to advertisements must be made in writing before the closing date. "Publisher" in this "General Conditions Document" refers to SYS-CON Publications, Inc. This index is provided as an additional service to our readers. The publisher does not assume any liability for errors or omissions. This index is provided as an additional service to our readers. The publisher does not assume any liability for errors or omissions.



You don't have to mortgage the farm for a **web content manager** that's **Powerful** and **Affordable**.

Our newest partners:

CFMX Hosting, Fulgen Technology, Harbour Light Strategic Marketing and ipXperts

➤ **Call for more info:**
1.866.870.6358

www.BeSavvy.com
Savvy Software Inc.



macromedia®
ALLIANCE PARTNER

The Real Estate Sample Application PART 1

Building the search functionality with Flash Forms



By Laura Arguello & Nahuel Foronda

With the release of Macromedia ColdFusion 7 and the arrival of Flash

Forms, developers were presented with an alternative to HTML forms that offered them additional functionality, such as full-featured controls not available in HTML and built-in validation.

That alone made Flash Forms appealing – and with the addition of pieces of ActionScript code, developers were able to create truly responsive forms. But because they were meant to be compatible with HTML forms, they still shared the same submit-refresh model. What if you could “submit” the form and, without a page refresh, get feedback from the server?

Enter Flash Remoting. In this series of tutorials you will learn how to create an application that allows users to search and retrieve records from a database, and then edit, add, and remove them from the database – all in one screen. Those functions will be presented in the context of a sample application, a real state management system that administers listings of properties for sale.

In Part 1 of this series, you will build the search

This article originally appeared on the Adobe Developer Center. Reprinted with permission.

functionality in the application.

Requirements

To complete this tutorial you will need to install the following software and files:

- **ColdFusion MX 7.01**

For a trial download go to http://www.macromedia.com/cfusion/tdrc/index.cfm?product=coldfusion&promoid=devcenter_tutorial_product_090903. To buy go to http://www.macromedia.com/software/coldfusion/buy/?promoid=devcenter_tutorial_coldfusion_090903. To get the updater go to <http://www.macromedia.com/support/coldfusion/downloads/updates.html#mx7>.

- **Database:** Microsoft Access, SQL Server, or MySQL
- **Tutorials and sample files:** <http://download.macromedia.com/pub/developer/realestate.zip>. To install the sample application on your computer, unzip the files, create a data source called realEstate in the ColdFusion Administrator, and then browse to the directory after you place the files. Read the full instructions in the Readme file.

Prerequisite Knowledge

Basic knowledge of ColdFusion components and Flash Forms and the ability to set up a data source and write simple SQL statements.

Overview of the Real Estate Management System Sample Application

This article focuses on the search functionality of the sample application (see Figure 1).

When you open the application, a small panel on the left lets users search for a property by specifying search criteria, such





as number of bedrooms or price range. The matching properties appear in a data grid in the upper-right panel, and when a user selects a property, the details appear in the lower-right panel (see Figure 2).

In the following sections you will do the following:

1. Create the search form.
2. Write a component to make the search query.
3. Create a Flash Remoting service that handles the search request.
4. Call the Flash Remoting service and show the results in a data grid

Creating the Search Form

The whole application user interface is one Flash Form. As such, its contents are within the cfform tag:

```
<cfform format="flash" name="RealEstateAdmin">
<!-- form content --->
</cfform>
```

The only necessary attribute of the cfform tag is the format="flash" attribute to create a Flash Form. By assigning a name to the form, you will have a named scope that you can use later. You can also set the form's dimensions by using the width and height attributes.

Note: If you are familiar with the process of defining a simple Flash Form, you can skip this section.

The Search panel (see Figure 3) contains several controls, all of them enclosed in a cfformgroup tag with a type attribute of "panel":

```
<cfformgroup type="panel" label="Search" height="440" >
<!-- controls --->
</cfformgroup>
```

These are the controls contained within the Search panel:

- The cfselect tags for Price range, Bedrooms, Bathrooms, and Minimum footage that create pop-up menus. You can populate cfselect tags from option tags, queries, or a combination of the two, as shown in this example:

```
<cfselect name="search_priceRangeFrom"
query="priceRange"
display="label"
value="data"
queryposition="below">
<option value="0">No min.</option>
</cfselect>
```

This cfselect tag uses a query called priceRange that contains two columns – data and label – which populate the search_priceRangeFrom select control. In addition, an extra option not present in the query ("No min.") is manually added using an option tag. The name you give each control is important because later you will need to reference the control name when you submit the form.

- <cfinput type="text"> tag for "MLS number", as follows:

```
<cfinput name="search_mls_id" type="text" />
```

- <cfinput type="radio"> tags for "Status." You need one cfinput tag for each type of listing status (Active, Sold, Back Up Offers, New Listing). Option button tags that belong to the same group must have the same name. See the following:

```
<cfinput type="radio" name="search_status" value="active" label="Active"/>
<cfinput type="radio" name="search_status" value="backUp" label="Back Up
Offers"/>
```

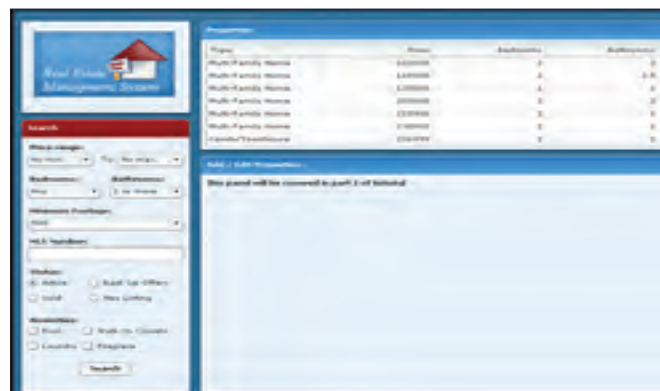


Figure 1: Overview of the real estate application

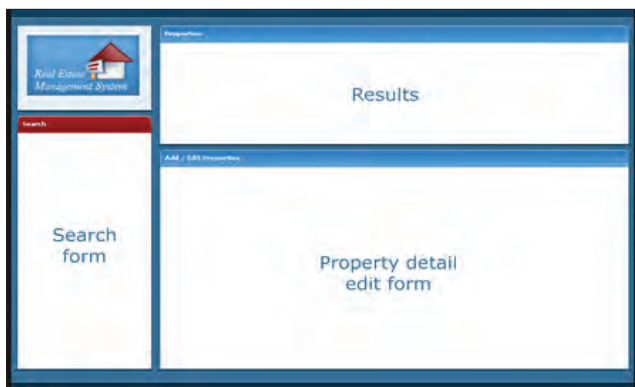


Figure 2: Layout of the application

```
<cfinput type="radio" name="search_status" value="sold" label="Sold"/>
```

- `<cfinput type="checkbox">` tags for "Amenities." You need one tag for each type of amenity (Pool, Laundry, Walk-in Closets, Fireplace). Unlike option buttons, each tag must have a different name, as follows:

```
<cfinput type="checkbox" name="search_hasPool" label="Pool"/>
```

```
<cfinput type="checkbox" name="search_hasLaundry" label="Laundry"/>
```

```
<cfinput type="checkbox" name="search_hasFireplace" label="Fireplace"/>
```

- `<cfformitem type="text">` tags for additional labels or titles, as follows:

```
<cfformitem type="text" style="fontWeight: bold;">Status:
</cfformitem>
```

- `<cfinput type="button">` tag for the Search button:

```
<cfinput type="button" name="search_submitBtn" value="Search" />
```

You must also use additional `cfformgroup` tags to lay out the controls. Look at the `index.cfm` source file in the ZIP download to see the complete code.

This small section of the application does not do anything yet. It only accepts user input of standard form controls: text inputs, option buttons, check boxes, and selects. At this point, you would normally add a Submit button to submit the form for processing on the server. But don't do that yet – wait just a moment and complete the following sections.

Coding the Search Query

As you may expect, you must write ColdFusion code to search the database.

The sample files contain a ColdFusion component (CFC), `components/ListingGateway.cfc`, that queries the database but you could use a custom tag as well.

The search in this component has several arguments that correspond to the different search criteria: status, price, pool and other amenities, and so forth. It contains a simple query using the `cfquery` tag. All the arguments have a default value and the SQL statements will vary each time a user specifies a different combination of arguments. For instance, if the user doesn't enter a status, the CFC ignores that criteria; therefore, the SQL statement will not contain that column in its WHERE clause. To exclude it, we used a simple `cfif` tag (see Listing 1).

When the query finishes processing, the CFC returns the results to the caller through the `<cfreturn listingQuery />` tag.

So far, you should be familiar with the methods used. You created a form to request data from the user and wrapped a query in a CFC to retrieve data from a database. You may be wondering how to connect the form and the CFC. Read onward.

Creating the Flash Remoting Service

Flash Remoting is a Macromedia technology that lets Flash Player make asynchronous calls to Web services and receive responses back from the call. During these calls, it can send and receive sets of data. It consists of a gateway that sits on the server and some components that are included in the SWF file. Fortunately, ColdFusion 7 already contains the Flash Remoting gateway, so you don't have to install any additional software. The SWF files generated by Flash Forms also contain the Flash Remoting components because they are used internally.

Just as Web services do, you must place Flash Remoting services in a Web-accessible directory. To create such a service, you use a CFC, writing the functions that you want to provide to the service consumers. These functions must have their access attribute set to **remote** in order to be accessible to Flash consumers.

Here is a simple example of a service:

```
<cfcomponent>
<cffunction
name="search"
access="remote"
returntype="query"
output="false"
hint="Returns listings matching the search criteria">
```

```
    <cfargument name="status" type="string"
default="" />
```

```
<!-- code here that generates a query called
myQuery -->
```

```
<cfreturn myQuery/>
```

Figure 3: The Search panel


```
</cffunction>
</cfcomponent>
```

We could have included the code that queries the database in the service CFC but it is better to separate presentation from data layers and keep well-defined tasks encapsulated in different components. In this way, the core components that access the database can be completely separate from service components that may access them. The service components, knowing that they are services, can offer additional functionality specifically targeted to Flash, such as formatting.

There is another advantage to this approach. It is common practice to keep components instantiated in a scope that spans several requests, such as the application or session scopes. When a Flash Remoting call is made to a component, however, this component is instantiated at that time – and every time a request is made, a new instance of the component is created.

This means that Flash Remoting cannot call an already instantiated component residing in memory in a shared scope. The only way to resolve this issue is to have a service component that differs from the component that does the actual work. When Flash Remoting calls the service component, this can reference the component residing in memory and make the appropriate invocations. For simplicity's sake, this example instantiates ListingGateway.cfc but the source files use the memory-resident

CFC approach (see Figure 4).

Because the Search panel sends simple data types that match the types expected by the ListingGateway component, the service can simply pass the same arguments along by using an argumentCollection (see Listing 2).

Name this component **ListingService.cfc** and save it in the **services** folders.

Calling the Service

Now that you have created a service, you are ready to call it. Provided that you are running ColdFusion on your local machine, the Flash Remoting gateway is typically located at <http://localhost/flashservices/gateway>. That is its default address for ColdFusion installations.

Please note that this address might be different if you are using the built-in Web server option (if your Web root is located at <http://localhost:8500>).

Adding ActionScript to the Flash Form

To call a service from a Flash Form, you must write some ActionScript. You can add any piece of ActionScript code to the form by writing inline statements in the controls' event handlers, such as onClick, or by writing code blocks in functions using the `<cfformitem type="script">` tag that has been added in the ColdFusion 7.01 Updater. For readability



One little add-on. A whole lot of control.

The ColdFusion Development and Support Console.

To learn more about FusionReactor and download your free trial version visit www.fusion-reactor.com



Trademarks and Registered Trademarks are the property of their respective owners.



Figure 4: Flash Remoting architecture using a tiered approach

and easier maintenance, we recommend using functions.

Because you will use this Flash Remoting service several times, we recommend that you store it in a variable that can be used by other functions and controls. Wrap the Flash Remoting setup code that creates the service and stores this variable in a called `setUpRemoting()`:

```
<cfformitem type="script">
  setUpRemoting():Void{

}
</cfformitem>
```

Inside the `cfformitem` tag, declare the connection and service proxy as local variables:

```
//note the gateway address
var connection:mx.remoting.Connection =
mx.remoting.NetServices.createGatewayConnection("http://localhost/flash-
services/gateway/");

var myService:mx.remoting.NetServiceProxy;
```

Declare an object that handles all the responses:

```
var responseHandler:Object = {};
```

This object must implement the necessary functions that will be automatically called when the server sends a response from a CFC. The default functions called on this object are `onResult` and `onStatus`. The server calls `onResult` when the service sends a successful response and calls `onStatus` when the service throws an error or when some other type of error occurs. That means that the `responseHandler` object must have those two functions if it is expected to act upon service responses:

```
responseHandler.onResult = ( results: Object ):Void {
  //handle service response
}

responseHandler.onStatus= ( status: Object ):Void {
  //handle error
}
```

Once the `responseHandler` objects knows how to behave, the actual service can be instantiated and stored in a global variable (`RealEstateAdmin.myGlobalObjects`), as follows:

```
RealEstateAdmin.myGlobalObjects.listingService = connection.
getService("RealEstate.services.ListingService", responseHandler );
```

The `getService()` method of the connection class takes two parameters; the first is the path to the service. To get the path to your service, find the directory structure from your Web root to your CFC, replace the slashes with dots, and remove the CFC extension.

For instance, if you saved the sample files in a folder called `RealEstate` in your Web root and saved `ListingManager.cfc` in a folder called `services`, the path would be `/RealEstate/services/ListingManager.cfc` from a browser. Translating that to dot notation becomes `RealEstate.services.ListingManager`.

The second parameter is the object that handles the responses, which is the object you just created: `responseHandler`. Note that the service is not actually created until you call a method on the service. So if you enter an incorrect path, you will not know it until you try to call it.

Now you have a method that sets up a Flash Remoting service, but if you never call this, the service will never be set up. You can use the `onload` attribute of the `cfform` tag, which was introduced with the ColdFusion 7.01 Updater to call the service, as follows:

```
<cfform name="RealEstateAdmin" format="flash" onload=" setUpRemoting()">
```

That is not enough, however, because the service is stored in a global variable that you must also declare as the form loads. You can create yet another to be called as the form loads that sets up this global variable and possibly other settings. While you are at it, call the `setUpRemoting()` right after that, as follows:

```
<cfform name="RealEstateAdmin"
format="flash"
onload="formOnLoad();">

<cfformitem type="script">

  formOnLoad():Void{
    //declare global variable
    RealEstateAdmin.myGlobalObjects = {};

    //call set up
    setUpRemoting();

    //set other properties as needed
  }
  //rest of the code
```

Assigning the Flash Remoting service to a global variable is only a suggestion. The most important code is what's inside the `setUpRemoting()`.

Finally, you are ready to call the service. Your Search panel readily accepts user input. You have set up the Flash Remoting service and your CFC is eagerly waiting to be called. Remember how the Search panel had a Search button? Now you can make it functional by adding some actions to its `onClick` attribute:

```
<cfinput type="button" name="searchSubmit" value="Search"
```



```
onclick="submitSearch()" />
```

But wait a minute. What is the submitSearch()? Because it is not built-in, you're going to create it next.

Creating the submitSearch()

Within the same <cfformitem type="script"> tag, add the new:

```
submitSearch():Void{  
  
}
```

The purpose of the submitSearch() is to gather all the information entered in the Search panel and send it to the ListingManager service.

If you recall, the CFC has several arguments. When making the call, you could send each argument, in order, as parameters to the call. But that becomes error-prone if the CFC has many arguments. An easy way to send several parameters is to use a structure with keys that have the same name as the arguments.

To create a structure, you instantiate an empty object (or structure) that will be the container for all the data entered in the Search panel form fields. Then, one by one, take the data entered in each field and assign it to a key in the empty object, as shown in Listing 3.

Once you have gathered all the necessary data for the structure, you can make the call to your CFC, sending all the parameters as one structure. Recall that "search" was the name of the service in the ListingManager CFC:

```
RealEstateAdmin.myGlobalObjects.listingService.search(searchArgs);
```

After you call the service and the matching records are retrieved, the server will send a query back with the results, even if it is an empty query. The responseHandler object handles the response in its onResult, but if you go back to the "Calling the Service" section, you'll see that the query was empty, which doesn't help much – when the results are returned, they will just be ignored.

Adding a Grid for the Results

The sample application shows the search results in a datagrid next to the Search panel. To implement this functionality, add a grid using cfgrid and name it listingGrid:

```
<cfgrid name="listingGrid" rowheaders="false">  
  <cfgridcolumn name="price" header="Price" />  
  <cfgridcolumn name="bedrooms" header="Bedrooms" />  
  <cfgridcolumn name="bathrooms" header="Bathrooms" />  
  <cfgridcolumn name="footage" header="Footage" />
```

SiteExecutive[™] Web Content Management

Empower your users to take control of the web

Hundreds of web sites and thousands of people use SiteExecutive to create and manage critical content across commercial, higher education, healthcare and government.

Create and Publish content
Empower non-technical users
Leverage ColdFusion
Automate content migration
Accelerate your business

SYSTEMSALLIANCE
Think Big. Work Smart



Consolidate your infrastructure and increase performance by leveraging the SunFire[™] x64 Server Family, powered by multi-core or single-core AMD Opteron[™] processors running Solaris[™] OS, Linux or Windows.

Visit www.siteexecutive.com/cfdj or call 877-797-2554

Sun, Sun Microsystems, the Sun logo, Solaris, and Sun Fire are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. SiteExecutive and the SiteExecutive logo are registered trademarks of Systems Alliance, Inc.

```
<cfgridcolumn name="type" header="Type" />
</cfgrid>
```

Notice that the headers of visible columns are properly named and the name attribute of each cfgridcolumn tag corresponds to a query column name.

Now that you have a control, assign the results to the grid's data provider:

```
var listingGrid = listingGrid;
responseHandler.onResult = (results: Object):Void {
    listingGrid.dataProvider = results;
}
```

Because the CFC might throw an error, you should also complete the onStatus to handle error responses:

```
responseHandler.onStatus = ( stat: Object ):Void {
    alert("Error: " + stat.description);
}
```

Dealing with Asynchronous Processes

If the service returns a query, you might be wondering why you can't simply use the following code:

```
//wrong code
listingGrid.dataProvider = listingService.search(searchArgs);
```

The above code is not correct because the call to search is done asynchronously. As soon as the call is triggered, it does not wait for the server's response; it continues running code written below it. For this reason, you must create an object (called responseHandler here) with functions that will be called when the server sends a response.

Another point to consider is that any communication across a network is unreliable. If the returned dataset is large, it might take some time to load. When using the page refresh model, the browser takes care of adding a progress bar to indicate that the next page is loading. Although this is not perfect, it does offer some feedback to the user. Now that your

application does not use a page refresh model, how do you indicate to users that their request is being processed and they have to wait?

There are several approaches to this. A simple method is to show the default clock cursor that Flash Forms use when they load data. This is a simple solution that is consistent with the other functions of Flash Forms. To add that functionality, insert the following snippet before making the call to the remote service in index.cfm:

```
mx.managers.CursorManager.showBusyCursor();
```

When the results come back, remove the cursor:

```
mx.managers.CursorManager.removeBusyCursor();
```

Add this snippet to both the onResult and onStatus functions because you don't know which one will get called.

Where to Go from Here

In this tutorial you learned the basic steps for creating a Flash Form application that takes advantage of Flash Remoting to provide a one-screen interface.

In Part 2, you will add functionality to retrieve, edit, and delete records in the database through Flash Remoting.



About the Authors

Laura Arguello is one of the founders of Blue Instant (www.blueinstant.com), where she has been creating Web applications for the last five years. Apart from her company, she also maintains a blog, AS Fusion (www.asfusion.com), where she and Nahuel Foronda write about ColdFusion, Flash, and other Macromedia and Web technologies.

Nahuel Foronda is the founder and lead Flash developer for Blue Instant (www.blueinstant.com), a Web development firm specializing in Rich Internet Applications. During his five years of experience with Flash, he has created award-winning applications and Web sites. He also maintains a blog on Flash and ColdFusion called AS Fusion (www.asfusion.com).

Listing 1

```
<cfcomponent name="ListingGateway">

<cffunction name="search" access="public" returntype="query"
output="false">
<cfargument name="status" type="string" default="" />
<cfargument name="priceFrom" type="numeric" default="0" />
<cfargument name="priceTo" type="numeric" default="0" />
<cfargument name="hasPool" type="boolean" default="false" />
<!--- additional arguments removed --->

<cfset var listingQuery = "" />

<cfquery name="listingQuery" datasource="realestate">
    SELECT *
    FROM listing
    WHERE
```

```
listing.price >=
<cfqueryparam value="#arguments.priceFrom#" cfsqltype="CF_SQL_MONEY"/>

<cfif arguments.priceTo GT 0>
    AND listing.price <=
<cfqueryparam value="#arguments.priceTo#" cfsqltype="CF_SQL_MONEY"/>
</cfif>

<cfif len(arguments.status)>
    AND listing.status =
<cfqueryparam value="#arguments.status#" cfsqltype="CF_SQL_VARCHAR"/>
</cfif>

<cfif arguments.hasPool>
    AND listing.hasPool = <cfqueryparam value="true"
cfsqltype="CF_SQL_BIT">
</cfif>
```



```
<!-- additional sql removed -->
```

```
ORDER BY price, listedOn  
</cfquery>
```

```
<cfreturn listingQuery />
```

```
</cffunction>
```

```
</cfcomponent>
```

Listing 2

```
<cffunction name="search"  
access="remote"  
returnType="query"  
output="false"  
hint="Returns listings matching the search criteria">  
<cfargument name="status" type="string" default="" />  
  <cfargument name="priceFrom" type="numeric" default="0" />  
  <cfargument name="priceTo" type="numeric" default="0" />  
  <cfargument name="hasPool" type="boolean" default="false" />  
  <!-- additional arguments removed -->  
  
  <!-- you could also use cfinvoke or access a component in memory  
  -->  
  <cfset var listingGateway =  
    createObject("component",  
      "RealEstate.components.ListingGateway") />  
  
  <cfreturn  
    listingGateway.search(argumentCollection=arguments) />  
</cffunction>
```

Listing 3

```
//get all the search criteria items  
var searchArguments = {};  
  
//in order to get data contained in Text inputs,  
//use myTextInputName.text  
searchArguments.mls_id = search_mls_id.text;  
  
//to get data from radio buttons,  
//use myRadioButtonName.selectedData  
searchArguments.status = search_status.selectedData;  
  
//checkboxes store their true/false value in their selected property,  
//myCheckboxName.selected  
searchArguments.hasPool = search_hasPool.selected;  
searchArguments.hasWalkInClosets = search_hasWalkInClosets.selected;  
searchArguments.hasLaundry = search_hasLaundry.selected;  
searchArguments.hasFireplace = search_hasFireplace.selected;  
  
//to get the selected choice in a dropdown,  
//use mySelectName.value or mySelectName.selectedItem.data  
searchArguments.footage = search_footage.value;  
searchArguments.bedrooms = search_bedrooms.value;  
searchArguments.bathrooms = search_bathrooms.value;  
searchArguments.priceFrom = search_priceRangeFrom.value;  
searchArguments.priceTo = search_priceRangeTo.value;
```

Download the Code...
Go to www.coldfusionjournal.com

Develop Powerful Web Applications with ColdFusion MX 7

- ▶ Easily build and deploy multi-step data entry forms with CFML tags
- ▶ Solve business reporting problems with Integrated Reporting
- ▶ Manage your account with HostMySite's Control Panel
- ▶ Call 24x7x365 for expert ColdFusion support

Visit hostmysite.com/cfdj for a special offer

 **HostMySite.com**
1-877-215-4678

Building a Drag-and-Drop Shopping Cart with AJAX

Creating an interactive shopping experience



By Joe Danziger

Keeping up with the latest Web technologies is tough nowadays. Every week it seems new sites are launched that push the envelope further and further in terms of what can be accomplished using just a Web browser.

The rise of AJAX over the past several months has taken over the development world and breathed new life into the Web. Although these techniques have been possible for many years now, the maturity of Web standards like XHTML and CSS now make it a viable alternative that will be viewable by all but the oldest browsers.

It's also been possible to accomplish many of the same things

using Flex or Flash, but the development cycle with those applications is typically more involved and the overhead often not justified.

We're going to harness the power of the Script.aculo.us JavaScript library to provide our interaction. As their Web site states, this library "provides you with easy-to-use, compatible and, ultimately, totally cool JavaScript libraries to make your web sites and web applications fly, Web 2.0 style." We're also going to utilize the <CF_SRS> library to handle the actual AJAX data piping to our application. Both of these libraries are free for all to use, and they're easier to integrate than you would think.

For this article, we'll create an interactive shopping experience allowing us to add items to our shopping basket by dragging and dropping them onto an icon of a shopping cart. We'll add AJAX functionality, allowing us to update our shopping cart without redrawing the entire screen. To save the trouble of setting up a product database, we'll use Amazon Web Services to search for DVDs and use those to shop from.

Start with a blank `index.cfm` in your root directory. You'll need to visit <http://script.aculo.us/downloads> to download the latest distribution (they're nearing a final release for version 1.5 as of this writing). Copy the "lib" and "src" directories into your empty directory. You'll need all of the .js files so just copy over the entire directory in each case. Next, type the following lines into the `<head></head>` section of your page:

```
<script src="./lib/prototype.js" type="text/javascript"></script>
<script src="./src/scriptaculous.js" type="text/javascript"></script>
```

We'll need a search box to submit our query to Amazon:

```
<form action="index.cfm" method="post">
Search: <input type="text" name="keywords" size="20" />
<input type="submit" name="search" value="Go" />
</form>
```

The page will look for a `form.search` variable and run an Amazon search when it is defined. Each item returned will be placed in its own styled div that will be able to be picked up and dragged.

The Scriptaculous library makes it easy to create "draggables" (the only required argument is the ID of the object that you want draggable). Listing 1 contains the code to search Amazon and return the results as draggable divs.

At this point, all of the items returned from the search will be in their own box and should be draggable around the screen. When we created each draggable, we set `"revert=true"`, which will snap each object back to its original location if not placed directly on a drop zone.

Next, we'll add a graphic of a shopping cart to our page, which will become a drop target on which to drag items. The Scriptaculous library also makes it easy to create these "droppables". The syntax is simply:

```
Droppables.add('id_of_element',[options]);
```

The code below creates a droppable zone of id "cart1" and also runs a function `onDrop()` that pops up an alert box letting the user know an item has been added. We then hide the element from view, which allows the other divs to slide over and adjust accordingly.

```

<script language="javascript" type="text/javascript">
Droppables.add('cart1', { onDrop:function(element) {
alert('Added UPC ' + element.id + ' to your shopping cart. ');
Element.hide(element.id);}});
</script>
```

The items should now be disappearing when dropped onto the shopping cart, but there's nothing going on behind the scenes yet. Now it's time to add some AJAX to process our shopping cart.

Although there are several AJAX libraries to choose from, we're going to use the ColdFusion Simple Remote Scripting `<CF_SRS>` library made available free of charge by Matthew Walker of ESWsoftware in New Zealand. `<CF_SRS>` uses an `IFRAME` for communication and encapsulates all of the dirty work for you. This library was chosen for its ability to handle HTML tables well and for its ability to interact directly with the browser's Document Object Model

(DOM) to output our shopping cart rows.

We'll start with an empty cart by including the following code:

```
<fieldset style="width:400px;">
<legend>Your shopping cart</legend>
<table border="0" cellspacing="0" cellpadding="5" id="tableCart">
<thead></thead><tbody></tbody></table>
<button onclick="emptyCartButton_onClick()" id="emptyCartButton">Clear
Shopping Cart</button>
</fieldset>
```

(Don't worry about the fact that our table body (`<tbody></tbody>`) is empty right now – we'll be populating it in just a second through AJAX.)

Next, you'll need to download the `<CF_SRS>` package from <http://www.eswsoftware.com/products/download/>. Copy the `srs.cfm` file into your Webroot (or you can add it to your CustomTags directory if you plan to do more AJAXing). You'll also need to create a subdirectory to hold the gateway pages that handle our AJAX data passing. Name the directory "SRS" and copy the `Application.cfm` and `OnRequestEnd.cfm` files into there from the "serverpages" directory in the zip file. You can use either regular CFM files or CFCs for these gateway pages (the download provides examples of each). The main thing to remember is that these pages should always return their results to "request.response".

Simply adding a `<cf_srs>` call to your page will handle the creation of the hidden `IFRAME` for you. Another great feature of the `CF_SRS` library is the ability to view an inline debugging window right inside the page you are working on. This allows you to see all of the data being passed back and forth through the gateway. You can enable this debugging by calling the tag as `<cf_srs trace>`. This line can be placed anywhere but we'll add it at the very end of the file.

Next, we'll need to create some JavaScript functions to handle the AJAX interactions. Add an `onLoad` function to your body tag as such: `<body onload="body_onLoad()">`. This will execute `body_onLoad()` when the page loads and we'll use this function to set up our gateway. The function should read as follows:

```
function body_onLoad() {
// create an SRS gateway to the cart.cfm page
objGateway = new gateway("srs/cart.cfm?");
// update cart in case of return visit
// code for this function is below
updateCart();
}
```

Once you have created your gateway, you can invoke the methods below to send requests to the server:

- ***objGateway.setListener(str)***: Use this method to specify the name of the function in your Web page that will handle the server's response. `str` is a string representing the function's name. The listener defaults to "alert", which will pop up a JavaScript `alert()` box containing the server's response. Note that while ColdFusion is a case-insensitive language, JavaScript is case-sensitive. If you return a structure to your listener function, all the structure keys will be rendered in JavaScript as lowercase.
- ***objGateway.setArguments(obj)***: Set the arguments and values to pass to the server. `obj` is an object literal, which is basi-

cally just a set of one or more attribute/value pairs wrapped in curly braces. Here's an example: { name:'Joe', age:30, country:'US' }. You can see that string values need to be wrapped in quotation marks, and colons (:) are used in place of equals signs (=).

- ***objGateway.resetArguments()***: Remove all the arguments previously set.
- ***objGateway.request()***: Send the request to the server.

Note that you can chain these methods together. For example, it is perfectly acceptable to write:

```
objGateway.resetArguments().setArguments( { state:'NY' } ).request()
```

Using what we know now, let's take another look at our `updateCart()` function that we're calling on `Load`.

```
function updateCart() {      objGateway.setListener('cartPacket_onRe-
ceive').setArguments( {action:'getCart'} ).request(); }
```

The function chains together several commands. It sets the listener to “cartPacket_onReceive”. That means that we’ll execute this JavaScript function whenever data is returned from our gateway. This function handles the generation of our table body that contains our cart rows (see Listing 2).

In our `updateCart()` function, we're also passing in an argument: `action=getCart`. This is going to be passed through to our `cart.cfm` gateway page. The full text of the gateway page is displayed in Listing 3.

We're passing in the action variable with a value of "getCart". This gets passed to our cart.cfm gateway page and causes the user's session cart to be returned as a query object. Whenever we need to update our cart to add or delete rows, we'll set our listener to 'cartPacket_onReceive' and then redraw the table body.

When we created our shopping cart on screen, we added the following button to clear our cart:

```
<button onclick="emptyCartButton_onClick()" id="emptyCartButton">Clear  
Shopping Cart</button>
```

We'll add two JavaScript functions to go along with that button. The first will confirm the delete and the second will issue a call to remove the items and redraw the cart:

```
function emptyCartButton_onClick() {
    if ( confirm('Are you sure you want to empty your cart?' ) )
clearCart();
}
```

```
function clearCart() {
    objGateway.setListener('cartPacket_onReceive').setArguments(
    {action: 'clearCart'} ).request();
}
```

Finally one more JavaScript function to be called when adding items to our cart:

```
function addToCart(upc) {
    objGateway.setListener('cartPacket_onReceive').setArguments( {
action:'addToCart',upc:upc } ).request();
}
```

Now that we have our `addToCart()` function coded, add the line `"addToCart(element.id);"` right before the `Element.hide` call in the shopping cart droppable. This will execute our `addToCart()` function and redraw the shopping cart when an item is dropped onto it.

And that's all there is to it! With just 150 lines of code, we were able to create an interactive, drag-and-drop shopping experience that many did not think was possible using just the browser. 🚀

About the Author

Joe Danziger is the founder and president of DJCentral.com, an online promotional tool for disc jockeys and other members of the electronic dance music industry. He has been developing professional ColdFusion solutions for over six years since version 1.5.

danziger@yahoo.com

Listing 1: Run Amazon search and return results:

```
<cfif isDefined("form.search")>
<!--- Submit REST query --->
<cfhttp url="http://webservices.amazon.com/onca/xml?Service=AWSECom
merceService&AWSAccessKeyId=#REPLACE_WITH_YOUR_KEY#&Operation=ItemS
earch&Keywords=#form.keywords#&SearchIndex=DVD&Sort=relevancerank&R
esponseGroup=Small,ItemAttributes,Images,Offers" throwOnError="yes"
charset="UTF-8"></cfhttp>
<!--- parse XML document --->
<cfset myXMLdoc = XmlParse(cfhttp.filecontent)>
<CFSET xnSearch = myXMLdoc.xmlRoot>

<cfoutput>
<!--- if results LT 10, loop through those, otherwise show first 10
--->
<cfloop index="i" from="1" to="#IF(xnSearch.Items.TotalResults.XMLText
```

```
gt 10,'10',xnSearch.Items.TotalResults.XMLText)#">
```

[illegible]

```
<!-- make this item draggable // reference by UPC as div id -->
```

```

<script type="text/javascript">
new Draggable('#xnSearch.Items.Item[i].ItemAttributes.UPC.XmlText#',
{revert:true});
</script>
</cfif>
</cfloop>
</cfoutput>
<br clear="all" />
</cfif>

```

Listing 2:

function cartPacket_onReceive(packet) { // generates an HTML table of cart items

```

    var theTable = document.getElementById('tableCart');
    var tbody = document.createElement("tbody");
    var tr, td;

    if ( packet.upc.length == 0 ) { // if no results just send back
empty cart
        tr = document.createElement("tr");
        td = document.createElement("td");
        td.colSpan = 4;
        td.innerHTML = "Your cart is empty.";
        tr.appendChild(td);
        tbody.appendChild(tr);
    }
    else
        for ( var i = 0 ; i < packet.upc.length; i++ ) { // loop and
create new row
            tr = document.createElement("tr");
            td = document.createElement("td");
            td.innerHTML = packet.qty[i];
            tr.appendChild(td);
            td = document.createElement("td");
            td.innerHTML = '';
            tr.appendChild(td);
            td = document.createElement("td");
            td.innerHTML = packet.title[i];
            tr.appendChild(td);
            td = document.createElement("td");
            td.innerHTML = packet.price[i];
            tr.appendChild(td);
            tbody.appendChild(tr);
        }

    theTable.replaceChild(tbody, theTable.childNodes[1]);
}

```

Listing 3: cart.cfm gateway page:

```

<cfswitch expression="#action#">

<cfcase value="getCart">
    <cfparam name="session.cart" default="#ArrayNew(1)#">
    <cfset cartQuery = QueryNew("qty,upc,title,image,price")>
    <cfloop from="1" to="#ArrayLen(session.cart)#" index="i">
        <cfset queryAddRow(cartQuery)>
        <cfset querySetCell(cartQuery, "qty", session.cart[i].qty,
i)>
        <cfset querySetCell(cartQuery, "upc", session.cart[i].upc,

```

```

i)>
        <cfset querySetCell(cartQuery, "title", session.cart[i].
title, i)>
        <cfset querySetCell(cartQuery, "image", session.cart[i].
image, i)>
        <cfset querySetCell(cartQuery, "price", session.cart[i].
price, i)>
    </cfloop>
    <cfset request.response = cartQuery>
</cfcase>

<cfcase value="addToCart">
    <cfparam name="session.cart" default="#ArrayNew(1)#">

<!-- get items from Amazon Web Services -->
<cfhttp url="http://webservices.amazon.com/onca/xml?Service=AWSECommerce
Service&AWSAccessKeyId=1MEQ9VMKAJS5A8DSHER2&Operation=ItemLookup&IdType=
UPC&SearchIndex=DVD&ItemId=#url.upc#&ResponseGroup=Small,ItemAttributes,
Images" throwOnError="yes" charset="UTF-8"></cfhttp>
<cfset myXMLdoc = XmlParse(cfhttp.filecontent)>
<CFSET xnProduct = myXMLdoc.xmlRoot>
<cfscript>
addItem = StructNew();
addItem.qty = 1;
addItem.upc = xnProduct.Items.Item.ItemAttributes.UPC.XmlText;
addItem.title = xnProduct.Items.Item.ItemAttributes.Title.XmlText;
addItem.image = xnProduct.Items.Item.SmallImage.URL.XmlText;
addItem.price = xnProduct.Items.Item.ItemAttributes.ListPrice.
FormattedPrice.XmlText;
</cfscript>
<cfset ArrayAppend(session.cart,addItem)>

<!-- return cart --->
<cfset cartQuery = QueryNew("qty,upc,title,image,price")>
<cfloop from="1" to="#ArrayLen(session.cart)#" index="i">
    <cfset queryAddRow(cartQuery)>
    <cfset querySetCell(cartQuery, "qty", session.cart[i].qty,
i)>
    <cfset querySetCell(cartQuery, "upc", session.cart[i].upc,
i)>
    <cfset querySetCell(cartQuery, "title", session.cart[i].
title, i)>
    <cfset querySetCell(cartQuery, "image", session.cart[i].
image, i)>
    <cfset querySetCell(cartQuery, "price", session.cart[i].
price, i)>
</cfloop>
<cfset request.response = cartQuery>
</cfcase>

<cfcase value="clearCart">
    <cfset ArrayClear(session.cart)>
    <cfset cartQuery = QueryNew("qty,upc,title,image,price")>
    <cfset request.response = cartQuery>
</cfcase>

</cfswitch>

```

Download the Code...
Go to www.coldfusionjournal.com

Toward a New Orthodoxy

Dynamic typing



By Hal Helms

Last month, we took a long look at strong typing. We saw that while strong typing offers many benefits in a language such as Java, trying to attach strong typing to ColdFusion produces really difficult problems.

And last month, due to my “in” connections, we were even able to briefly interview the Java compiler.

Some might have thought that I was arguing the case against strong/static typing, as many developers have recently. They have argued that static typing is an emperor without clothes. Bruce Eckel, of *Thinking in Java* fame, has called strong typing “a Faustian bargain.” Guido van Rossum, creator of Python, has said, “Strong typing catches many bugs, but it also makes you focus too much on getting the types right and not enough on getting the rest of the program right.” Developer Ned Batchelder has summed up the case against strong typing: Static typing prevents certain kinds of failures. Unfortunately, it also prevents certain kinds of successes.”

My own view of strong/static typing is different: I find it immensely helpful in a language like Java. I think that a good deal of the problems developers have with strong typing in that language would be greatly ameliorated by designing with interfaces. And I appear to be in good company. The “Gang of Four” authors who wrote the seminal work, *Design Patterns*, offered readers the excellent advice: “Design to interfaces, not to implementations.”

Clearly, the issue of strong versus weak / static versus dynamic typing is one on which many smart and experienced developers disagree. In this article, I want to make a case that, whatever your view of strong/static typing is, in regards to ColdFusion, it is a moot point simply because ColdFusion does not support strong typing.

But is this a defect in ColdFusion or merely a difference? Certainly languages such as Smalltalk, Python, and Ruby seem to do fine without strong typing. To that illustrious list, I would add ColdFusion.

In place of the debate over whether strong or weak typing is better, perhaps a better question is this: How can we benefit from the dynamic typing ColdFusion offers? First, I think we would do well to adopt what Dave Thomas (*The Pragmatic Programmers*) whimsically calls *duck typing* – as in, “if it walks like a duck and quacks like a duck, it must be a duck!”

With duck typing, the idea of a type becomes more a matter of “Can this object respond to this message?” rather than “Does this object fit into a type hierarchy?”. This is particularly important with ColdFusion since the only type hierarchy available is one of inheritance, which commonly causes fragile code.

Let’s look at an example of duck typing. Having just returned from teaching two classes in Las Vegas, I’ll use an example from the poker craze that has turned ordinarily calm souls into wild men (and women!) pushing their entire stack of chips into the pot while calling “All in!” Let’s suppose that we’ve been asked to come up with a domain model for a Las Vegas poker room.

We might begin with a Player CFC that has the following methods:

- bet (accepts a numeric value)
- check
- fold
- raise (accepts a numeric value)

We probably also want a Dealer CFC with these methods:

- dealHoleCards
- promptPlayer
- acceptBet (accepts a numeric value)
- dealFlop
- dealTurn
- dealRiver
- declareWinner

Perhaps, we’ll have an Employee class that Dealer inherits. No doubt an Employee will have many methods but the one closest to our hearts is this:

- acceptPay (accepts a numeric value)

Now, a little known fact is that many Las Vegas poker rooms employ proposition player (or simply props). Prop players are paid by the house to stimulate action, but play with their own money. Since prop players are paid by the house, our Prop class might rightfully extend Employee.

But, let’s suppose that we need to model a tournament. Our Tournament CFC could have methods like:

Efficient Web Content Management

CommonSpot™ Content Server is the ColdFusion developer's leading choice for efficient Web content management.

Our rich Web publishing framework empowers developers with out-of-the-box features such as template-driven pages, custom content objects, granular security, and powerful ColdFusion integration hooks (just to name a few), allowing you to rapidly build and efficiently deploy dynamic, high performance Web sites.

CommonSpot's open architecture and extensive APIs enable easy customization and integration of external applications. With CommonSpot, you can design and build exactly what you need. Best of all, CommonSpot puts content management in the hands of content owners. With non-technical users responsible for creating and managing Web content, developers are freed to focus on strategic application development.

Evaluate CommonSpot today.

To see your site *running* under CommonSpot, call us at 1.800.940.3087.



fast
easy
affordable

features.

- 100 % browser-based
- Content object architecture
- Template driven pages
- 50+ standard elements
- Content reuse
- Content scheduling
- Personalization
- Flexible workflow
- Granular security
- Mac-based authoring
- 508 compliance
- Full CSS support
- Custom metadata
- Taxonomy module
- Extensible via ColdFusion
- Web Services content API
- Custom authentication
- Replication
- Static site generation
- Multilanguage support

1.800.940.3087
www.paperthin.com

Paper | Thin

© Copyright 2005 PaperThin, Inc. All rights reserved.

Don't Miss
CFDJ's
Next
Issue!



Have you ever wanted to "see" what's going on inside your ColdFusion servers and applications?

Now you can.

Monitor and troubleshoot your ColdFusion applications and servers in real-time with SeeFusion.



Key Features:

- * View metrics for request times, queries, throughput, memory utilization, server uptime, and other critical data
- * Attach debugging output to pages
- * Log metrics to a database for in-depth analysis
- * Stop long-running/hung requests
- * Easily monitor multiple servers at once
- * Connect to notification systems via XML bridge
- * And more!

www.seefusion.com

SeeFusion is a product of Webapper Services, LLC - Development, Consulting, Products
www.webapper.com

- register (accepts a Player and a numeric amount)
- payWinners

We run into a problem here. It may be that in order to fill a tournament, the house asks a prop to enter a tournament. But the Tournament's register method accepts only Player objects. In a strongly typed system, we really want our Prop players to be of both Employee and Player types.

Java solves this problem through *interfaces*. A class can extend only one superclass, but it can *implement* as many interfaces as needed. Interfaces define a type and the methods that can be called on that type, but provide no method bodies – that is, no implementations of these methods. Assuming we want Prop to extend Employee (and that ColdFusion had interfaces), we could provide a Player interface that would be functionally similar to this Java version:

```
public interface Player{
    public void bet(int bet);
    public void check();
    public void fold();
    public void raise(int
bet);
}
```

Interfaces provide *type* inheritance, but not code inheritance: the implementing class must provide the method bodies for all methods defined in the interfaces it implements. While this does solve the problem of multiple-typing, it does

so at a cost. If we need to change the implementation of a Player, it must be done in both the Player class and the Prop class. This violates what is sometimes called the DRY (Don't Repeat Yourself) principle. According to Dave Thomas, "DRY says that every piece of system knowledge should have one authoritative, unambiguous representation." Here, we have (at least) two representations for each Player method.

For the many ColdFusion developers who dutifully provide type properties for their arguments and return type proper-

ties for their methods, this has caused ineluctable problems that have not been resolved. But again, let's ask the question: Is this a defect in ColdFusion or merely a difference?

If we embrace ColdFusion's weak typing, there is no problem. As long as both the Player object and the Prop object can respond to the methods needed to play in the tournament (bet, check, fold, and raise), our code will run without a snag. "But it's not type safe!" the strong type proponent might argue. That's true, but ColdFusion doesn't offer type safety (nor do Smalltalk, Python, or Ruby). We might say that if an object can bet, check, fold, and raise like a Player, then it is a Player – regardless of its class.

Welcome to duck typing. All that is required is that we give up the illusion of type safety (and with it, the return-type and type properties of methods and arguments, respectively). And there are other benefits to duck typing. One informal study found that Java applications compared to the same applications in Smalltalk required between 85–230%



more code. That's a lot of finger typing.

Of course, speed doesn't mean much if we are writing fragile code that will be hard to maintain. Since the overwhelming majority of the cost of code over its lifetime is spent on maintenance, anything that compromises the maintainability of code must be highly suspect. And strong typing must be credited with catching many errors that may not be at first apparent, but which will surface over an application's lifetime.


As you might guess, the duck typers have an answer for this: *unit testing*. Unit

testing is a part of a movement/philosophy known as test-driven development. In *test-driven development*, automated tests are written as part of the software design cycle. Such tests act like "mini clients," exercising the various methods of each class to uncover any bugs.

For such testing to be feasible, it must be automated. Kent Beck and Erich Gamma produced a tool to do just this for Java programmers: JUnit. It has since been implemented in many languages. ColdFusion has two programs that provide automated test unit functionality, CFCUnit (www.cfcUnit.com) and CFUnit (www.cfunit.sourceforge.net). I strongly encourage readers to explore the difference using such tools can have on the quality of your software.

Perhaps you noticed that something was left out of our duck typing solution to the problem of prop players. If both the Prop and the Player classes have to independently implement the various player methods, are we not still in violation of the DRY principle? We are. The solution to this problem comes from applying the notion of *mixins* to ColdFusion.

Mixins are a technique, first introduced in a Lisp language variant known as *Flavors*, in which a class defines the attributes of the class, but does not define the methods of the class. Instead, the methods are defined in separate files known as mixins. In our example, both the Prop and the Player class would define their own instance variables, but would include the methods for bet, check, fold, and raise from a separate file – perhaps PlayerMixin.

The good news is that ColdFusion makes the use of mixins very simple and straightforward. Next month, we'll look at code that makes use of mixins and we'll see how mixins can either be applied to an entire class or to individual objects. 

About the Author

Hal Helms is the author of several books on programming. Hal teaches classes in Java, C#.NET, OO Programming with CFCs, Design Patterns in CFCs, ColdFusion Foundations, Mach-II, and Fusebox. He's the author of the popular Occasional Newsletter and his site is www.halhelms.com.

hal@halhelms.com



Visit the *New*

www.SYS-CON.com

Website Today!

The World's Leading i-Technology News and Information Source

24/7

FREE NEWSLETTERS

Stay ahead of the i-Technology curve with
E-mail updates on what's happening in your industry

SYS-CON.TV

Watch video of breaking news, interviews with industry leaders, and how-to tutorials

BLOG-N-PLAY!

Read web logs from the movers and shakers or create your own blog to be read by millions

WEBCAST

Streaming video on today's i-Technology news, events, and webinars

EDUCATION

The world's leading online i-Technology university

RESEARCH

i-Technology data "and" analysis for business decision-makers

MAGAZINES

View the current issue and past archives of your favorite i-Technology journal

INTERNATIONAL SITES

Get all the news and information happening in other countries worldwide

JUMP TO THE LEADING i-TECHNOLOGY WEBSITES:

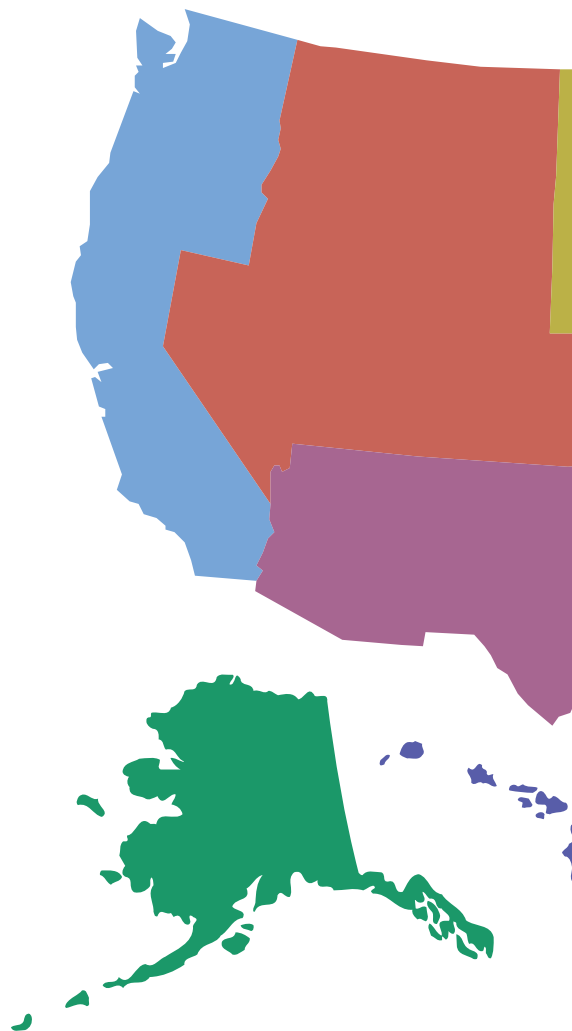
<i>IT Solutions Guide</i>	<i>MX Developer's Journal</i>
<i>Information Storage+Security Journal</i>	<i>ColdFusion Developer's Journal</i>
<i>JDJ</i>	<i>XML Journal</i>
<i>Web Services Journal</i>	<i>Wireless Business & Technology</i>
<i>.NET Developer's Journal</i>	<i>Symbian Developer's Journal</i>
<i>LinuxWorld Magazine</i>	<i>WebSphere Journal</i>
<i>Linux Business News</i>	<i>WLDJ</i>
<i>Eclipse Developer's Journal</i>	<i>PowerBuilder Developer's Journal</i>

ColdFusion

For more information go to...

U.S.

Alabama Huntsville Huntsville, AL CFUG www.nacflug.com	Colorado Denver Denver CFUG www.denvercfug.org/	Iowa Johnston Des Moines, IA CFUG www.hungrycow.com/cfug/
Alaska Anchorage Alaska Macromedia User Group www.akmmug.org	Delaware Kennett Square Wilmington CFUG www.bvcfug.org/	Kentucky Louisville Louisville, KY CFUG www.kymug.com/
Arizona Phoenix www.azcfug.org	Delaware Laurel Delmarva CFUG www.delmarva-cfug.org	Louisiana Lafayette Lafayette, LA MMUG www.cflib.org/acadiana/
Arizona Tucson www.tucsoncfug.org	Florida Jacksonville Jacksonville, FL CFUG www.jaxfusion.org/	Maryland Lexington Park California, MD CFUG http://www.smdcfug.org
California San Francisco Bay Area CFUG www.bacflug.net	Florida Winter Springs Gainesville, FL CFUG www.gisfusion.com/	Maryland Rockville Maryland CFUG www.cfug-md.org
California Riverside Inland Empire CFUG www.sccfug.org	Florida Plantation South Florida CFUG www.cfug-sfl.org	Massachusetts Quincy Boston, MA CFUG www.bostoncfug.com
California EL Segundo Los Angeles CFUG www.sccfug.org	Florida Tallahassee Tallahassee, FL CFUG www.tcfug.com/	Michigan East Lansing Mid Michigan CFUG www.coldfusion.org/pages/index.cfm
California Irvine Orange County CFUG www.sccfug.org	Florida Palm Harbor Tampa, FL CFUG www.tbmmug.org	Minnesota Brooklyn Park Twin Cities CFUG www.colderfusion.com
California Davis Sacramento, CA CFUG www.saccfug.org	Georgia Atlanta Atlanta, GA CFUG www.acfug.org	Missouri Overland Park Kansas City, MO CFUG www.kcfusion.org
California San Jose (temporary) Silicon Valley CFUG www.siliconvalleycfug.com	Illinois East Central East Central Illinois CFUG www.ecicfug.org/	Missouri O'Fallon St. Louis, MO CFUG www.stlmmug.com/
California San Diego San Diego, CA CFUG www.sdcfug.org/	Indiana Avon Indianapolis, IN CFUG www.hoosierfusion.com	New Jersey Princeton Central New Jersey CFUG http://www.cjcfug.us/
California Long Beach Southern California CFUG www.sccfug.org	Indiana Mishawaka Northern Indiana CFUG www.ninmug.org	Nevada Las Vegas Las Vegas CFUG www.sncfug.com/
		New York Albany Albany, NY CFUG www.anycfug.org
		New York Brooklyn New York, NY CFUG www.nycfug.org
		New York Syracuse Syracuse, NY CFUG www.cfugcny.org
		North Carolina Raleigh Raleigh, NC CFUG www.ccfug.org
		Ohio Dayton Greater Dayton CFUG www.cfd Dayton.com
		Oregon Portland Portland, OR CFUG www.pdxcfug.org



User Groups

<http://www.macromedia.com/cfusion/usergroups>



Pennsylvania
Carlisle
Central Penn CFUG
www.centralpenncfug.org

Pennsylvania
Exton
Philadelphia, PA CFUG
www.phillycfug.org/

Pennsylvania
State College
State College, PA CFUG
www.mmug-sc.org/

Rhode Island
Providence
Providence, RI CFUG
www.ricfug.com/www/meetings.cfm

Tennessee
LaVergne
Nashville, TN CFUG
www.ncfug.com

Tennessee
Germantown
Memphis, TN CFUG
www.mmug.mind-over-data.com

Texas
Austin
Austin, TX CFUG
www.cftexas.net/

Texas
Corinth
Dallas, TX CFUG
www.dfwcfug.org/

Texas
Houston
Houston Area CFUG
www.houcfug.org

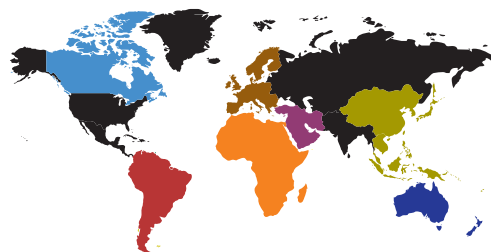
Utah
North Salt Lake
Salt Lake City, UT CFUG
www.slcfug.org

Washington
Seattle
Seattle CFUG
www.seattlecfug.com

About CFUGs

ColdFusion User Groups provide a forum of support and technology to Web professionals of all levels and professions. Whether you're a designer, seasoned developer, or just starting out – ColdFusion User Groups strengthen community, increase networking, unveil the latest technology innovations, and reveal the techniques that turn novices into experts, and experts into gurus.

INTERNATIONAL



Australia
ACT CFUG
www.actcfug.com

Australia
Queensland CFUG
www.qld.cfug.org.au/

Australia
Southern Australia CFUG
www.cfug.org.au/

Australia
Victoria CFUG
www.cfcentral.com.au

Australia
Western Australia CFUG
www.cfugwa.com/

Italy
Italy CFUG
www.cfmentor.com

Japan
Japan CFUG
cfusion.itfrontier.co.jp/jcfug/jcfug.cfm

Scotland
Scottish CFUG
www.scottishcfug.com

South Africa
Joe-Burg, South Africa CFUG
www.mmug.co.za

South Africa
Cape Town, South Africa CFUG
www.mmug.co.za

Brazil
Brasilia CFUG
www.cfugdf.com.br

Brazil
Rio de Janeiro CFUG
www.cfugrio.com.br/

Brazil
Sao Paulo CFUG
www.cfugsp.com.br

Canada
Kingston, ON CFUG
www.kcfug.org

Canada
Toronto, ON CFUG
www.cfugtoronto.org

Ireland
Dublin, Ireland CFUG
www.mmug-dublin.com/

Spain
Spanish CFUG
www.cfugspain.org

Switzerland
Swiss CFUG
www.swisscfug.org

Thailand
Bangkok, Thailand CFUG
thaicfug.tei.or.th/

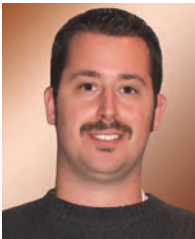
Turkey
Turkey CFUG
www.cfrtr.net

United Kingdom
UK CFUG
www.ukcfug.org



BrowserHawk 9 by cyScape

Customize your Web visitor's site experience



By Nicholas Tunney

I have been developing Web applications for years, and have been using random JavaScript snippets gleaned from the Web to test a user's browser and configured properties. Since this script would be run client side, it required the

user to have JavaScript enabled and didn't always work in all browsers.

I had always wished for a quick and easy server-side solution to handle simple browser and plug-in checking, but had never even fathomed having access to the amount of information that is made available using BrowserHawk from cyScape. If you think this is just another browser detection script, you could not be more wrong. Not only does BrowserHawk detect basic browser settings, it also detects disabled plug-ins, blocked pop-ups, AJAX support, blocked ports, estimated download time, WAP settings, and so much more! BrowserHawk Enterprise can actually detect over 125 properties and settings of your visitor's browser and connection.

Installation

I downloaded and installed BrowserHawk 9.0.0.21 Enterprise edition from the cyScape Web site. You can choose from either an FTP or HTTP download. The download was small and quick, even using my hotel's wireless connection. The version that works with ColdFusion is written in Java, and I installed and tested this on the CF 7.0.1 standalone server. As a side note, BrowserHawk is also available for ActiveX and .NET.

There is surprisingly quite a bit of documentation that comes with BrowserHawk;

however, some of the initial installation instructions might be a bit complicated for beginners. Unfortunately, some links in the documentation are broken, and the online documentation does not seem to cover the Java installation; however, scrolling down through the documentation, you will find instructions for several different configurations that are much clearer than the initial documentation. There currently is no installer for the Java version, but the rumor at cyScape is that the installer is an important addition that they are including in the next version. It seems that ColdFusion is a big priority on their list. They have also hinted at some additional exciting ColdFusion-specific features, but you'll have to wait to hear about that from them.

Since this is a Java install, the server needs to be restarted at the end of the installation process, but if you followed the instructions correctly, you will be able to test BrowserHawk immediately with the detailed example files that are included in the Zip file. The demo files are all linked through a single index.html page, which makes it very easy to navigate and test with no additional setup.

Usage

Initially, as I am like most developers and just dive right in, I began using the JavaBean immediately instead of reading the help files. It was very straightforward to initialize by looking through the sample code, and after dumping the object and viewing what functions and values are available, I was able to

detect some pretty neat settings. I then read through the readme file a bit and found that cyScape has included a ColdFusion custom tag that makes it very simple to use most functionality in BrowserHawk. This tag acts as a wrapper for the BrowserHawk JavaBean and will be a great help for ColdFusion beginners and for new users learning BrowserHawk. The custom tag does not immediately accept all of the properties available, but the document states that it does return results for the most popular BrowserHawk tests including detecting browser type, version, platform, broadband vs. dialup connections, Flash and Acrobat plug-ins, disabled



Subscribe Today!

SAVE 16%

12 Issues for **\$89⁹⁹**

OFFER SUBJECT TO CHANGE WITHOUT NOTICE



- Exclusive feature articles
- Latest *CFDJ* product reviews
- Interviews with the hottest names in ColdFusion
- Code examples you can use in your applications
- *CFDJ* tips and techniques

That's a savings of \$29.89 off the annual newsstand rate. Visit our site at www.sys-con.com/coldfusion or call 1-800-303-5282 and subscribe today!

ColdFusion Developer's Journal



PRODUCT FEATURE	STANDARD	PROFESSIONAL	ENTERPRISE
Detect all IE, Netscape, and Mozilla browsers	✓	✓	✓
Detect hundreds of other browsers including WebTV, Opera, Gecko based browsers, and AOL	✓	✓	✓
Detect hundreds of search engines and crawlers	✓	✓	✓
Detect installed Microsoft .NET CLR and version	✓	✓	✓
GUI based Editor for viewing/customizing browser definitions	✓	✓	✓
Integration with Microsoft InterDev and other IDEs	✓	✓	✓
Detect 80+ properties including platform, DHTML, XML, StyleSheets, FileUpload, and more	✓	✓	✓
Reverse DNS lookups	✓	✓	✓
Detect WAP and PDA devices	✓	✓	✓
Detect SSL key size (i.e. 40 Vs. 128 bit) from HTTPS pages	✓	✓	✓
Integration with CountryHawk (requires CountryHawk, available separately)	✓	✓	✓
Detect disabled cookies		✓	✓
Detect disabled images		✓	✓
Detect when IE is run in High Security mode		✓	✓
Automatic browser definition updates		✓	✓
Detect disabled JavaScript and disabled Java applets		✓	✓
Detect Flash, Shockwave, MediaPlayer, RealPlayer, Acrobat, Quicktime, Adobe SVG, and Authorware plug-ins		✓	✓
Detect screen size, browser window size and color depth		✓	✓
Detect broadband Vs. dial-up connections		✓	✓
Detect user connection speed throughput in bits per second		✓	✓
Detect browser local time and time zone difference between client and server		✓	✓
Detect language of user and OS		✓	✓
Easily generate XML representation of BH results		✓	✓
Detect Font Smoothing setting in control panel		✓	✓
DashCache for maximum performance		✓	✓
Automatic caching and retrieval of BH results to the user's session (Automatic Session Caching)		✓	✓
Real-time, unique browser statistics not available with any other web reporting software			✓
Automatic logging of all statistics to your database with no programming required!			✓
Detect blocked popups!			✓
Detect installed fonts			✓
Detect disabled ActiveX controls and disabled VBScript			✓
Detect disabled/blocked SSL			✓
Detect JVM version and Java vendor			✓
Detect build number of the installed Microsoft JVM dll			✓
Detect service pack (i.e. "SP2;Q271292") and browser build (i.e. 6,0,2462,0)			✓
Detect detailed operating system version information, including Mac OS X versions			✓
Enhanced connection speed testing to finely control the balance between testing time and accuracy of results			✓
Download time estimator automatically calculates the download time for a specified file based on user's connection speed			✓
Detect JavaScript and VBScript build numbers (i.e. 5.6.6012)			✓
Detect SSL key size without requiring a separate HTTPS page for testing			✓
Detect firewalls, proxy settings, and blocked ports			✓
Detect iPIX, Crystal Reports, Citrix ICA, Viewpoint, MapGuide, and Java plug-ins			✓
Detect custom plug-ins			✓
Detect View->Text Size setting in IE 5+			✓
Detect minimum Windows Installer version			✓
Detect Microsoft XML parser version			✓
Detect Microsoft NetMeeting build number			✓
Real-time monitoring of internal DashCache parameters for optimizing component speed			✓

Table 1:

cookies/JavaScript, and much more. Also, it seems very simple to configure the custom tag to use the remainder of the BrowserHawk methods.

The custom tag simply takes a comma-delimited list of the properties you want to test for and returns a structure of name/value pairs.

Example

```
<cfmodule template="BrowserHawk4J.
cfm" propstotest="Browser, Platform"
bhvarname="myResults" />

<cfoutput>
    #myResults.Browser#, #myResults.Platform#
</cfoutput>
```

On my system, this returned "FireFox, WinXP". The process is also very fast, and auto-caching is an available feature in BrowserHawk. This seems to be especially useful if you are performing some of the more complex tests like client port detection (that's right, client port detection! I told you this package is powerful).

In Table 1 I have included the features comparison across the three available versions of BrowserHawk. This grid will show you the unbelievable power possible in this package. You can quickly see this is so much more than those old JavaScripts we have become accustomed to.

Reporting

BrowserHawk also allows you to store detected settings in a database and generate reports. I know this is something many of my clients have requested over the years, and BrowserHawk makes it simple. The reporting package includes SQL scripts to create the necessary table for persisting this information in Microsoft SQL, DB2, MySQL, Oracle, and Apache Derby. I did not test the reporting feature on my server as I did not have enough data to make it interesting, but I have included a sample screenshot (see Figure 1). From the demo it seems as though by default you can report on about 50 different statistics.

Issues

One thing I was disappointed with is that the online demo of BrowserHawk (located at <http://www.cyscape.com/>

2005 – A Year in Review

The “Web 2.0” buzz began in 2005 and people are into it. Macromedia even went so far as to show off what type of Web 2.0 applications will be possible with Flex 2.0 next year, and what an impressive demonstration it was. AJAX became popular, and has given developers a library of functionality that allows Web 2.0 style clients to be built and delivered to clients on a variety of browsers. Of course, Macromedia’s public announcement of Flex 2.0 and FlexBuilder 2 (along with a ColdFusion connector) are of major importance. I like AJAX and I think it’s a great idea – especially for developers who are not allowed to use Flash as the platform for their user interfaces. Personally, I’m much more excited about Flex 2.0 – it has a wider reach than AJAX, allows developers to do things that AJAX simply cannot do, and in my opinion is more developer-friendly than AJAX. Macromedia released their Macromedia Labs site – finally they have joined the ranks of companies like Microsoft and Google in offering the public a forum for looking at technologies while they are still in the research and development phases. It is there, at Macromedia Labs (<http://labs.macromedia.com>), that the public alpha of FlexBuilder 2 was released. Toward the end of 2005 I also announced that **CFDJ** will begin to offer more articles on Web 2.0 concepts and, more important, on techniques for building better applications using AJAX, Flex, and ColdFusion.

The last, but certainly not least, significant happening of 2005 is the announcement and finalization of Adobe’s acquisition of Macromedia. Those of us who remember the Macromedia acquisition of Allaire know that these things do mean change, but typically the changes are for the better. With the acquisition of Macromedia by Adobe Systems, Adobe not only owns and controls the best digital document format for content and best graphics and layout tools on the planet, but is also the proud parent of the best format, tools, and server products for rich content, animation, and business interfaces on the Web, the best online collaboration server, and the best platform for developing Web applications. Having all of these products under one umbrella creates opportunities for every product that were unfathomable before. Adobe also made a point of stressing the value they place on the Macromedia user communities, and appear eager to begin working with us to strengthen the popularity of the products as well as strengthening the communities themselves. Adobe was very quick, through interviews, press releases, and letters to the community, to send a warm welcome to us all. I’d like to conclude this editorial, and 2005, by reciprocating to any Adobe employee who this article finds its way to: so many of us look forward to seeing what you do with the products we use and love, and we look forward to working with you as well. I sincerely can’t wait to see what 2006 has in store for us all...

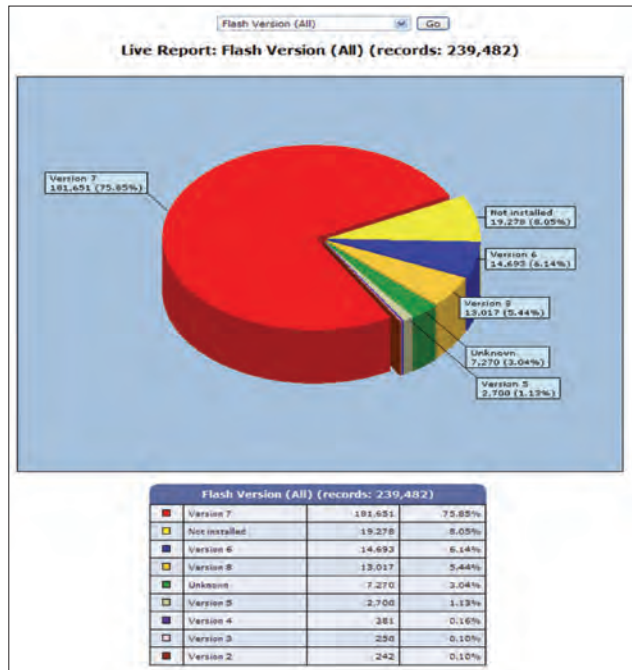


Figure 1

[showbrow.aspx?bhcp=1](#)) shows the visitor’s country as an accessible setting; however, Enterprise edition of BrowserHawk does not include this feature. You need to take a look at CountryHawk for that awesome functionality, although the Enterprise edition does include many other location specific values. Also, I could not find a list of all possible properties I could return. The readme file includes two links that apparently list all properties, but the links are dead. I’m sure cyScope will get these links up and working soon.

Conclusion

I personally think that we will be hearing more about this powerful product. It is definitely the software to use for customizing your Web visitor’s site experience. I definitely will be using BrowserHawk in the future. While you may not need all of the functionality of the Enterprise edition, they have a fair pricing structure that will allow you to purchase only the functionality you need. BrowserHawk is already being used by many high profile companies such as Verizon, Intel and even Adobe. Go download the demo at www.cyscope.com today.

About the Author

Nicholas Tunney is a Macromedia Certified ColdFusion developer, and has been programming ColdFusion for over 7 years. He is currently Senior Software Architect for AboutWeb, a consulting firm located in Rockville, Maryland. To learn more about ColdFusion, visit Nic’s Blog at <http://www.nictunney.com>.

ntunney@aboutweb.com

Event Gateways

An exciting and powerful feature

By Ben Forta &
Sarge Sargent

ColdFusion MX 7 has some exciting new functionality but the most important and revolutionary of these new features, hands down, is the event gateway. Event gateways are to application server environments what the CFQUERY tag was to database interaction.

As you will see, the event gateway technology transforms ColdFusion MX 7 (CFMX 7) from just another Web application server to an enterprise services platform, allowing CFMX 7 applications to work with just about any other application and/or platform over any well-defined protocol including SMS, RMI, MQ, JSM, AMS, TCP, and UDP. If you have ever worked with message-oriented middleware, much of the paradigm behind event gateways will be obvious to you. On the other hand, if you've not, the ColdFusion event gateway operates like MOM.

Event gateways also allow ColdFusion applications to perform in new and nontrivial ways. You, the developer, can move away from an HTTP-based request/response development approach to event-based frameworks that respond to events created by such things as a folder change or a phone call. Before you begin this article, take a breath and clear your mind. Try to forget what you know about traditional Web development. Read this article with an open mind, knowing that there is no possible way a single article can cover everything that event gateways can do. The only limit now to ColdFusion MX 7 is your imagination. Okay, let's begin.

What Is an Event Gateway?

ColdFusion MX 7 gateways are Java classes that implement an application programming interface (API) provided with the CFMX7 application server. Figure 1 shows a high-level diagram of the event gateway communications. Event gateways communicate with the ColdFusion event gateway services through CFEvent objects, which we will define later. The event gateway service system puts CFEvent objects, if necessary, in a queue

and then passes them to the ColdFusion run-time engine for processing as input to ColdFusion Components (Listener CFCs). The Listener CFC might return output to the event gateway services subsystem and then back to the event gateway.

You create gateway instances from a gateway type. Instances correspond to individual copies of a gateway that are running. Gateway instances are Java objects that are started/stopped through the ColdFusion Administrator. Each gateway instance specifies a CFC to handle incoming messages. You can have more than one instance of an event gateway type, and each instance will have its own configuration. For example, you can have multiple instances of a given gateway type, each with a different login, phone number, buddy name, directories to watch, and so forth.

Simply put, ColdFusion's new event gateway exposes the power of J2EE's underlying messaging technology. This allows you to

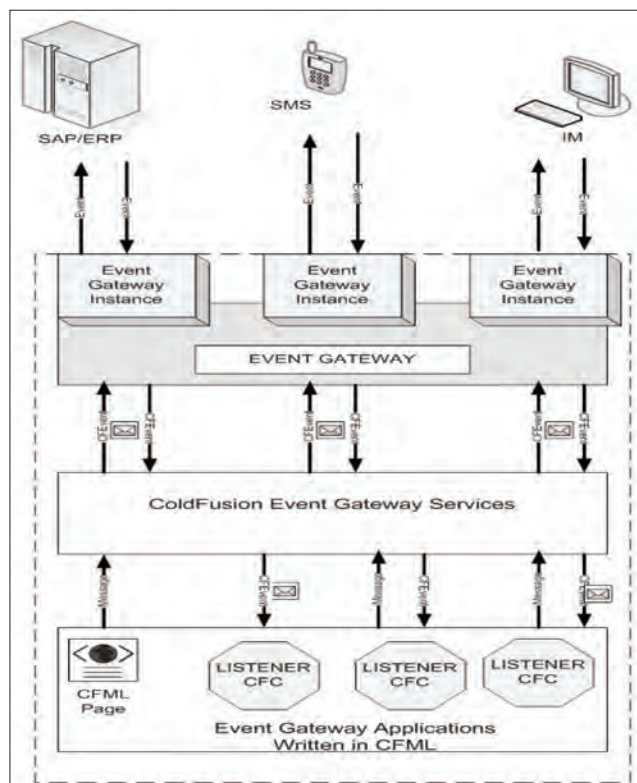


Figure 1: ColdFusion event gateway communications

This article is an excerpt from *Advanced Macromedia ColdFusion MX 7 Application Development* by Ben Forta and Sarge Sargent. Reprinted with permission.

use ColdFusion in a way that's different from the traditional HTTP-based request/response model. Web applications have grown more sophisticated and moved from isolated information-retrieval applications to mission-critical applications that not only expose some applications to users but tie together older processing systems, databases, enterprise resource-planning systems, and so on. As this evolution progressed, developers frequently had to kludge special and proprietary connections to other applications. ColdFusion developers often found it hard to connect to legacy applications, ERP systems, package applications, and other commercial off-the-shelf applications – usually by using a variety of other languages and proprietary integrations tools.

This difficult task was compounded by previous ColdFusion versions sometimes having difficulty scaling. Requests to these applications often hold open a connection and/or thread while ColdFusion connects to another application, retrieves that information, and creates a response, even though no user or application is waiting for a response. For example, a scheduled task that fires off a batch process sending thousands of e-mails to subscribers could tie up ColdFusion threads for an hour until the process is finished. Now, with ColdFusion event gateways, you can fire an event from a ColdFusion page or CFC and run that batch process in the background, while keeping your ColdFusion threads free to handle Web requests.

Events in ColdFusion are asynchronous requests, which means that a request can occur but the actual requested action can occur at another time; this is different from traditional HTTP requests, which either are fulfilled or timed out. ColdFusion events can reside in a queue where each event is fulfilled when the server has capacity and resources to handle it. This is especially useful for applications in which a transaction or process doesn't need to happen at the time a user requests it, such as notifying a system that an order has been placed. Asynchronous messaging has many other uses as well.

Categories of Event Gateways

Event gateway systems essentially fall into two categories.



Initiators are CFMX7 applications that generate an event message from a CFC or CFML application page and send the message using an event gateway instance. An example of an initiator would be an application that checks a POP 3 server for new messages and, if new messages are there, sends an SMS notification that forwards the information to your cell phone. This is done by using the `sendGatewayMessage` function to send outgoing messages like these SMS text messages through an event gateway.

Responders are CFMX7 applications that receive an event message from some external source through the listening event gateway instance. The event gateway service then routes the event to a listener CFC that you configure when you create an event instance. Depending on the method triggered by the event, the CFC can return a response. An example could be an IM Help Desk application. Let's say you want to give users the ability to IM your company and get help on technical problems. You could set up an IM event gateway and instance, and have a listener CFC that waits for IMs

and then routes them to the first available technical support person or, if no one is free, sends a message asking the customer to wait. The customer is then put into a queue and later connected to the first available technical-support person.

A responder listener CFC listens for an event from a given gateway instance and processes the event structure passed to it, to return a re-

sponse. The event structure contains the message, along with some detail about its origin. If you dumped the `CFEvent` message, it might look something like Figure 2.

Creating a Simple Gateway Application

In Chapter 29, "Extending ColdFusion with Java," we created several examples for a photo album application. One of the examples we created looped over the contents of a directory and made thumbnails for all the images found in that directory. That's pretty cool, but it requires a user to interact with the

application through a Web browser. What if your boss now asks you to allow users to FTP-upload files of photos, and wants the application to detect that a file has been uploaded and create an associated directory of thumbnails! Normally you couldn't do this with ColdFusion because you're not interacting with the server directly through HTTP – but with ColdFusion MX 7 and the `DirectoryWatcher` event gateway, it's a snap.

For this example, the `DirectoryWatcher` event gateway (supplied with CFMX7) will create a simple responder application. The `DirectoryWatcher` gateway sends events to a listener CFC when a file is created, deleted, or modified in a directory you tell the gateway to watch. It checks the directory at an interval specified in a configuration file that you edit, and when the interval has passed, checks for changes since last time. If it finds any changes, `DirectoryWatcher` sends a message to a listener CFC, which can perform the unzipping and creation of thumbnails.

First you need to configure the gateway configuration file, found in:

```
coldfusion_root\gateway\config\directory-watcher.cfg
```

For this example we'll assume that your ColdFusion root is on drive C, so a literal example of the file path would look like this:

```
C:\CFusionMX7\gateway\config\directory-watcher.cfg
```

Open this file and edit the very first attribute, `directory`, and have it point to where you will FTP your zipped files. For this example, the directory is called `gallery` and the path is as follows:

```
directory=C:/inetpub/wwwroot/JavaInteg/imageio/resize/gallery/
```

Table 1 lists a number of other configuration file attributes you can set. After

CFMETHOD	onAdd	
CFCPATH	C:\inetpub\wwwroot\JavaInteg\imageio\resize\DirectoryWatcher.cfc	
DATA	FILENAME	C:\inetpub\wwwroot\JavaInteg\imageio\resize\gallery\29_01_sleepythread.tif
	LASTMODIFIED	{ts '2005-02-16 17:15:37'}
	TYPE	ADD
GATEWAYID	PhotoAlbum	
GATEWAYTYPE	FileWatcher	
ORIGINATORID	{empty-string}	

Figure 2: Detail of a `CFEvent` message

you've edited the directory path, save the file.

Note: For the directory gateway configuration file, you want to use the forward slash (/) rather than the normal backslash (\).

Note: Not all event gateways will have a configuration file, and each configuration file will be unique to that specific gateway.

Next, we'll create a CFC that listens for events from the gateway. For this example, we'll only use the `onAdd` method supplied by the `DirectoryWatcher` gateway. What we want our CFC to do is listen for an event from the directory watcher, saying that files have been added to the gallery directory. Then the CFC will call the `imagemanipulator.cfc` (from Chapter 29) to create the thumbnails. For this example we are assuming that the `DirectoryWatcher.cfc` is being placed in the same directory as the `imagemanipulator.cfc` which is `webroot\JavaInteg\` and the directory gallery is beneath that. Last,

we'll log the event. The code to do all this is in Listing 1.

Most of the code here is pretty straightforward. You'll see some things that are specific to working with event gateways. The first is that the listener CFC expects a struct called `CFCEvent`, which is a Java object that is mapped to a ColdFusion struct. The `CFCEvent` object contains a variety of information, including a `GatewayID`, `OriginatorID`, `GatewayType`, `CFCPath`, `CFCMethod`, `CFCTimeout`, and `Data`. Figure 2 shows an example of what the `CFCEvent` message might look like in our example. Table 2 describes each node in the structure.

In Listing 1, the `CFCEvent` message is used only to log the `DirectoryWatcher` method that was used, `data.type`, as well as the file and file path, `data.filename`. In a more complex application, you may want your CFC to take actions based on specific information from the `CFCEvent` message.

VALUE	REQUIRED OR OPTIONAL	DESCRIPTION
Directory	Required	Path to the directory to watch.
Recurse	Optional	Whether to check subdirectories. The default value is No.
extensions	Optional	Comma-delimited list of extensions to watch. The event gateway logs only changed files that have these extensions. An asterisk (*) indicates all files; this is the default.
Interval	Optional	Number of milliseconds between the event gateway's checks on the directory. The default value is 60 seconds.
addFunction	Optional	Name of the function to call when a file is added. The default value is <code>onAdd</code> .
changeFunction	Optional	Name of the function to call when a file is changed. The default value is <code>onChange</code> .
deleteFunction	Optional	Name of the function to call when a file is deleted. The default value is <code>onDelete</code> .

Table 1: Gateway Configuration File Attributes

FIELD	DESCRIPTION
GatewayID	The event gateway that sent the event or will handle the outgoing message. The value is the ID of an event gateway instance configured on the ColdFusion MX Administrator Gateways page. If the application calls the <code>SendGatewayMessage</code> function to respond to the event gateway, it uses this ID as the function's first parameter.
OriginatorID	The originator of the message. The value depends on the protocol or event gateway type. Some event gateways might require this value in response messages to identify the destination of the response. Identifies the sender of the message.
Data	A structure containing the event data, including the message. Contents depend on the event gateway type. This field corresponds to the <code>SendGatewayMessage</code> function's second parameter.
GatewayType	The type of event gateway, such as SMS. This field can be used by an application that can process messages from multiple event gateway types. This value is the gateway type name specified by the event gateway class. It is not necessarily the same as the gateway type name in the ColdFusion MX Administrator.
CFCPath	The location of the listener CFC. The listener CFC does not need to use this field.
CFCMethod	The listener method that ColdFusion invokes to process the event. The listener CFC does not need to use this field.
CFCTimeout	The timeout, in seconds, for the listener CFC to process the event request. The listener CFC does not need to use this field.

Table 2: CFCEvent Information

Now to get our code to actually work, we need to do two other things. The first is to go into the ColdFusion Administrator and create a mapping for the `DirectoryWatcher` CFC; otherwise, the event gateway will not know where to look. The second thing we need to do is create an instance of the event gateway.

Creating an Event Gateway Instance

Before you can use the example in Listing 1, you must create an instance of the event gateway.

First go to the ColdFusion Administrator and select `Event Gateways > Gateway Instances`. You'll see something like Figure 3.

You should now see a form with a number of fields including `Gateway ID`, `Gateway Type`, `CFC Path`, etc. To create the gateway follow these steps:

- The first field is the `Gateway ID`, which can be anything; for our example, let's just use `PhotoAlbum`.
- The next field is `Gateway Type`, which is a drop-down list of all the registered gateways. For this example, we select `DirectoryWatcher`, which watches a directory for file changes.
- The next field is `CFC Path`, which is the CFC path to our listener `DirectoryWatcher.cfc`. After that you need to add the path to the `directory-watcher.cfc`.
- Finally, select the `Startup Mode` drop-down list to choose whether you want the event instance to be started automatically, manually, or disabled on startup of ColdFusion. Choose `Automatic` and then click `Add Gateway Instance`.

You should now see your gateway instance in the `Configured ColdFusion Event Gateway Instances` area of the page. Be sure to click the green button to start your event gateway instance. Now your event gateway instance is running and ready to respond to events.

You can make as many instances as you want of a specific gateway, so you could create several `DirectoryWatcher` instances in order to watch many directories (although it would probably be better to just change the `DirectoryWatcher` class to support multiple directories). For each event gateway for which you want to create applications, you must have at least

BY NOW THERE ISN'T A
SOFTWARE DEVELOPER ON EARTH
WHO ISN'T AWARE OF THE
COLLECTION OF PROGRAMMING
TECHNOLOGIES KNOWN AS AJAX!

REAL-WORLD AJAX

ONE DAY SEMINAR

www.ajaxseminar.com

March 13, 2006

Marriott

Marriott Marquis Times Square
New York City

For more information
Call 201-802-3022 or
email events@sys-con.com

REAL WORLD

How, in concrete terms, can you take advantage in your own projects of this newly popular way of delivering online content to users without reloading an entire page?

How soon can you be monetizing AJAX?

This "Real-World AJAX" one-day seminar aims to answer these exact questions...

Led by "The Father of AJAX" himself, the charismatic Jesse James Garrett of Adaptive Path, "Real-World AJAX" has one overriding organizing principle: its aim is to make sure that delegates fortunate enough to secure themselves a place – before all seats are sold out – will leave the seminar with a sufficient grasp of Asynchronous JavaScript and XML to enable them to build their first AJAX application on their own when they get back to their offices.

HURRY!
LIMITED SEATING
THIS SEMINAR WILL
SELL-OUT
CALL 201-802-3022
TO REGISTER!



Jeremy Geelan
Conference Chair, Real-World AJAX
jeremy@sys-con.com



Jesse James Garrett
Father of AJAX



Scott Dietzen
Creator of WebLogic,
Ph.D., President and
CTO, Zimbra



Bill Scott
AJAX Evangelist
of Yahoo!



David Heinemeier Hansson
Creator of Ruby on Rails



Satish Dharmaraj
Father of JSP, Co-
Founder & CEO, Zimbra



Rob Gonda
Best-Selling AJAX
Author, CTO,
iChameleon Group



Dion Hinchcliffe
Co-founder & CTO,
Sphere of Influence Inc.



Ross Dargahi
Well-known AJAX
Evangelist & Co-founder
and VP of Engineering,
Zimbra

Early Bird	\$995
(Before January 31, 2006)	
Discounted Price	\$1,195
(Before February 28, 2006)	
Seminar Price	\$1,295
(After February 28, 2006, and if any seat is available)	

MEDIA SPONSOR



LIVE SIMULCAST!
AROUND THE WORLD ON SYS-CON TV

PRODUCED BY
SYS-CON
EVENTS

one instance actually running if you want to use it.

Now that you have the Directory-Watcher instance running, you can test the example by copying some images into the gallery folder. Then navigate to your ColdFusion logs directory; you should see a new log created, `watcher.log`, which is our listener CFC's log. If you have set your `directorywatcher.cfg` to 60 seconds, you may have to wait, but eventually you'll see the `watcher.log` update. Look there, and you'll see that it has recorded the action, add, and the file and file path of the images you added to the directory. Examine the thumbnails directory under gallery to see the thumbnails that have been created. At this point it is easy to add code to support things like unzipping zip files of images, moving them to their own new directory based on the zip file names, and then creating the thumbnails – or any other cool functionality you like.

Note: If you change or delete a file in the watched directory, you'll see an error in the `gateway.log` file and the `cfexception` log file. This is because we did not define these methods in our CFC. You can easily do this and just add these methods and have them do nothing, or log the change to a file, etc.

At this point you have created a simple responder application and have set up an event gateway instance. You've learned something about how ColdFusion event gateways work, but there is a lot more to event gateways. Now we'll explore them further, discussing initiator applications, some of the particular differences between coding CFML for gateways and

other applications, and how to debug your CFML gateway applications. Finally, we'll look at a simple example of creating your own custom gateway using ColdFusion MX 7's API for gateways using Java.

Creating an Initiator Application Using the ColdFusion Gateway

So far, you've seen how to create a responder application that listens for events from an event gateway instance and takes some sort of action. Now we'll study an application called an *initiator application* that sends a message to an event gateway. This example uses the ColdFusion Gateway to asynchronously log messages to a file via a simple CFC.

Logging may seem trivial, especially since you can just use CFLOG, but CFML pages that use CFLOG to write large amounts of data to a log file can seriously degrade an application's performance, as well as tie up threads that could be better used serving your application's Web users. In addition, you might have an application like a B2B that needs to log large amounts of information, such as every type of transaction between partners for legal reasons. Thus you might want to decouple logging into its own subsystem, not only for performance but for good design. Using the ColdFusion Gateway allows you to create applications that call CFCs asynchronously, which is perfect for this example. Okay, let's look at some code (see Listing 2).

As you can see, this is a very simple CFC that accepts a CFEvent message and logs information from it. After you save this file to a directory, create an event gateway

send a message to the event gateway that will call the `DataLogger.cfc` to log the event. We do this using `SendGatewayMessage` with "GatewayID" and a struct with an event message. Let's look at Listing 3.

We have just added some simple code in a CFSCRIPT block and then used the `SendGatewayMessage()` function, which takes two required attributes. The first is the Gateway ID and the second is data, a structure that conforms to the specific gateway type. In this case, the gateway expects the event to contain a message node as well as an optional file node.

As you can see, creating an initiator application is simple enough, and in this case we have a listener that both responds to an event from a gateway and fires an event to a gateway.

Debugging CFML Applications for Event Gateways

When you need to develop ColdFusion applications that use event gateways, you need to be particularly careful – CFCs that are responding to events work differently than when they are responding to a normal page request. If an event gateway triggers a CFC and that CFC throws an error, the event gateway continues to function without pause and does not display any sort of debugging information back to you. For this reason, you'll need to follow some different development paradigms – especially with regard to debugging, so that you can make sure your CFCs and event gateways are functioning as expected. In this section we'll examine some techniques you can and should use to help you debug and write better code.

The first technique is to make extensive use of CFTRY, CFCATCH, and CFLOG. Keep in mind that CFCs called by the event gateway will fail, but the event gateway will continue processing requests without returning anything to you. Catching any exceptions in your CFML and dumping them to a specific file will allow you to much more easily debug your applications.

You can also use CFDUMP in your application code and write the output of CFDUMP to a file. An easy way to do this is to use CFSAVECONTENT to wrap things like CFDUMP, loop over stack traces and so forth, and then put the CFSAVECONTENT variable in the `text=""` attribute of the `cflog` file.

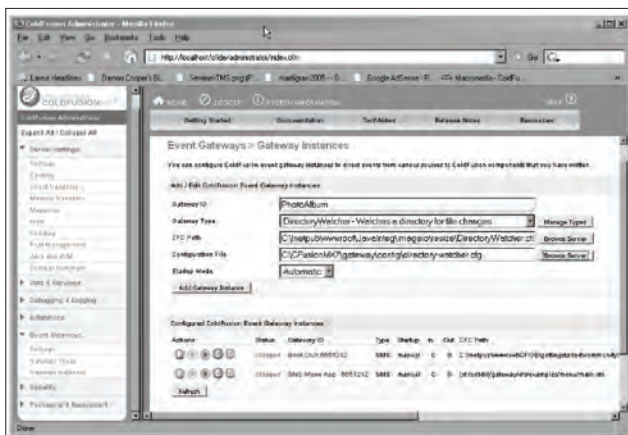


Figure 3: The Event Gateways > Gateway Instances configuration screen in the ColdFusion Administrator

instance in the Event Gateways page of the Administrator, as described earlier. For this example, give the instance a Gateway ID of `DataLogger` and point it to this CFC. Then leave the Configuration File field blank, create the instance, and start it running.

Now we can take our previous code and, instead of using CFLOG directly for logging information,

Tip: Using these techniques in Listings 2 and 3 to log and trap error information from your Gateway applications is an excellent way to add debugging/logging to your event gateway applications.

Something else you can do is to put debugging and tracking variables in your Application scope. Then you can dump the contents of the scope and see any information about what you are tracking.

Another major debugging approach that should always be part of any development – but especially with CFCs and CFML applications that use gateways – is creating unit tests. You can easily do this by creating simple CFML pages that use the `SendGatewayMessage` function to simulate a message from a CFC to the event gateway. We will discuss this shortly, in the section “Creating Your Own Custom Gateways,” as well as how to call those pages.

Finally, consider running the ColdFusion MX 7 server from the command line. If you do this, you can use `Java System.out.println` to dump error message information to your DOS or command shell. This technique is really useful when working with Java objects from ColdFusion MX 7. Simply add into your code something like this:

```
<cfscript>
sys = createObject("java", "java.lang.System");
```

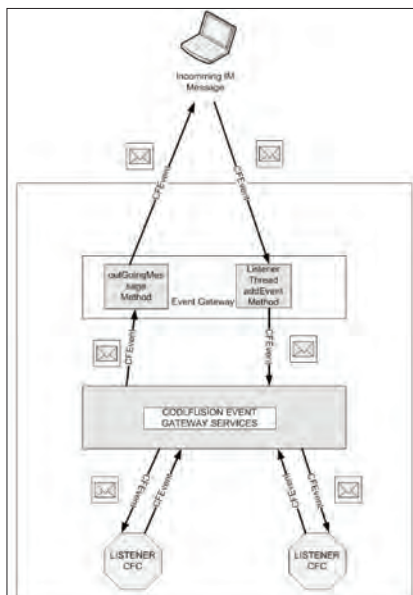


Figure 4: The flow of an Event through a Gateway application

DATA TYPE	METHOD	DESCRIPTION
void	setGatewayID(String id)	Sets the ID that uniquely defines the gateway instance.
void	setCFCListeners(String[] listeners)	Sets an array of CFC listeners.
GatewayHelper	getHelper()	Returns a gateway helper class (if there is one) so that a CFC can invoke gateway-specific functions that might be useful to the CFML developer.
String	getGatewayID()	Returns the gateway ID.
int	getStatus()	Gets the event gateway status, which can be either STARTING, RUNNING, STOPPING, STOPPED, FAILED.
void	start()	Starts the event gateway. ColdFusion calls this method on startup.
void	stop()	This method stops the event gateway and kills any threads and cleans up any resources it was using.
void	restart()	Restarts a running event gateway.
String	outgoingMessage (CFEvent cmesg)	Sends a message from the gateway to a resource.

Table 3: Methods of the ColdFusion Gateway Interface

```
sys.out.println("Debugging message goes here");
</cfscript>
```

Creating Your Own Custom Gateways

Although CFMX7 comes with a number of useful gateways, what happens when you want to connect to something that's not covered by one of the gateways provided? What if you want to connect to your MQ Series server or SAP via the BAPI messaging interface? The answer is to create your own gateway. Writing CFMX7 gateways is a fairly straightforward task, but gateways are developed completely in Java and you'll need to have a solid understanding of Java to write your own event gateways.

The ColdFusion Event Gateway Architecture

Event gateways listen for events and pass them to ColdFusion for handling by an application's listener CFC. The gateway does this by implementing the `ColdFusion.eventgateway.Gateway` interface and by using the `ColdFusion.gatewayServices` class.

Let's take a more detailed look at the overall architecture of the ColdFusion Gateway. In Figure 4 you can follow the path of a simple event through the system. Consider an incoming Instant Messaging event. The event gateway has a listener thread that will receive the message and call the Gateway Services `addEvent` method to send ColdFusion a `CFEvent` Hash Map, which will be converted into a ColdFusion structure.

Now let's say your ColdFusion event application sends an event pack to the same IM source from a CFC. The event from the CFC would be passed to the ColdFusion event gateway via the event

gateway's `outgoingMessage` method, which creates a `CFEvent` object with the appropriate destination and payload information.

The following sections introduce each of the major elements in constructing an event gateway.

Event Gateway Elements

There are six basic elements that are used to create and configure a ColdFusion Gateway: the Gateway interface, the `GatewayServices` class, the `CFEvent` class, the `GatewayHelper` class, the Gateway configuration class, and the Gateway development tools.

Note: The classes for the ColdFusion Gateway are in the `cfusion.jar`. Make sure when you compile your Java event gateways that you add the `cfusion.jar` to your classpath as well as any other `.jar` files you plan to use. The `cfusion.jar` file can usually be found in the `C:\CFusionMX7\lib` directory.

Gateway Interface

All ColdFusion event gateways have to implement the `ColdFusion.eventservice.Gateway` interface. Table 3 gives a list of the Gateway interfaces methods. You can also find this information in a handy Javadoc at `cf_root/gateway/docs/api/index.html`.

Gateway Services Class

To interact with the ColdFusion event gateway services, you used the Gateway class `ColdFusion.eventgateway.GatewayServices`. Table 4 lists the methods that the `GatewayServices` class implements. Like the Gateway interface, `GatewayServices` is summarized in the gateway Javadoc.

DATA TYPE	METHOD	DESCRIPTION
GatewayServices	getGatewayServices()	Returns a GatewayServices object.
int	getQueueSize()	Returns the current size of the gateway event queue.
int	getMaxQueueSize()	Returns the maximum size of the gateway event queue.
Logger	getLogger()	Gets the default event logging object.
Logger	getLogger(String logfile)	Gets a custom event logging object
boolean	addEvent(CFEvent msg)	Adds an event to the ColdFusion event service processing queue for delivery to a listener CFC.

Table 4: Methods of the GatewayServices Class

DATA TYPE	METHOD	DESCRIPTION
CFEvent	String gatewayID()	CFEvent constructor that expects a string, which is the gatewayID.
void	setGatewayType(String type)	Sets the type of event gateway, such as IM, SMS, or EMail.
void	setData(Map data)	Adds the gateway-specific data, including any message contents, as a Java Map to the CFEvent object.
void	setOriginatorID(String id)	Sets the originator of an incoming message.
void	setCFCPath(String path)	Specifies the listener CFC that will process this event.
void	setCFCMethod(String method)	Sets the name of the CFC method that should process an incoming message.
void	setCFCTimeout(String seconds)	Sets the timeout, in seconds, during which the listener CFC must process the event request before ColdFusion gateway services terminates the request and logs an error in the application.log file.
String	getGatewayType()	Identifies the type of the gateway from which this message originated.
Map	getData()	Gets the message contents and other gateway-specific information.
String	getOriginatorID()	Identifies the originator of an incoming message.
String	getCFCPath()	Gets the path to the listener CFC that processes this message.
String	getCFCMethod()	Gets the name of the CFC method that processes the message.
String	getGatewayID()	Identifies the event gateway instance, as specified in the ColdFusion Administrator.

Table 5: Methods of the CFEvent Class

CFEvent Class

As you have seen earlier, the CFEvent object is the container for the message passed to CFCs from gateways. Your gateway does this by using the GatewayServices.addEvent method to send an instance of the CFEvent object. Gateways receive CFEvents when ColdFusion calls a gateway's outgoingMessage method.

The CFEvent class uses java.util.Hashtable to create a HashMap that models your message. That HashMap converts the contents into case-insensitive information for consumption by ColdFusion (which is caseless). Table 5 relates the methods for the CFEvent class.

Gateway Helper Class

The GatewayHelper class provides an interface (Marker class) that can be used to mark a helper class that can be returned by a gateway. ColdFusion developers creating CFCs can use the functions Gateway.getHelper() and getGatewayHelper to invoke gateway-specific utility functions such as retrieving an IM phone book.

This class is returned by the CFML function getGatewayCFCHelper(gatewa

yID). CFCs cannot get a direct reference to the Gateway object in order to protect the gateway's key operations, such as start, stop, and restart. Look in cf_root\gateway\src\examples\socket\SocketGateway.java to see an example of how to implement a GatewayHelper.

Gateway Configuration Files

Depending on what your gateway will do, creating a configuration file can be very useful. The example in Listing 4, which we'll examine shortly, connects to a POP3 server to see if there are new messages on the mail server. Instead of hard-coding the hostname, login, and password for the mail account, we'll use a configuration file to store this information. To do this, you'll want to use the java.util.Properties to create a properties file using name=value pairs.

Note: The Sun Javadoc for the Properties class can be found here: <http://java.sun.com/j2se/1.4.2/docs/api/java/util/Properties.html>.

Gateway Development Tools

ColdFusion ships with several useful

tools to help you in developing your ColdFusion custom gateways. The first of these is a generic abstract gateway class from which you can derive your gateway class.

Another useful tool is the EmptyGateway class found in cf_root\gateway\src\examples\EmptyGateway.java. You can use this Java class as a template for building your own custom classes. Before reading the next section on the POP3 Custom Gateway, it's recommended that you review this class, as well as compare it to some of the example classes provided in the examples directory which can be usually found in your cf_root\gateway\src\examples.

If you are an Apache Ant user, look into the build.xml file found at cf_root\gateway\src\build.xml. This useful tool lets you build all the examples in the examples directory and contains the paths to all the .jar files needed by these examples. If you are an Eclipse user, right-click gateway/src/build.xml and Run to compile the examples.

All the CFCs to create gateway instances to deploy and test the examples are found in cf_root\gateway\cfc. Also, the configuration file for any of the examples can be found in cf_root\gateway\config. You can use these configuration files as samples for making your own. The ColdFusion MX 7 documentation has even more comprehensive information on the gateway classes and tools available to you.

A POP3 Custom Gateway

You've read about the architecture and classes that make up the event gateway, so let's go ahead and walk through making our own custom gateway. In this example, you're going to make a simple gateway that connects to a POP3 server and, if there are new e-mails on the POP server for a specific account defined in a configuration file, the gateway will send an event to a listener CFC with the number of new e-mails.

To work through this example, you'll need to have the J2EE.jar, to make use of the javax.mail classes, and the cfusion.jar in your classpath. If you are using Ant you can modify the build.xml file that can usually be found at cf_root\gateway\src\ and use that. When you do compile the example, you'll want to compile the code and then deploy it as POP3Gateway.jar along with the J2EE.jar. Now let's look at Listing 4.

eclipse) developer's journal

A DIGITAL PUBLICATION

Your One-Stop Guide to the Eclipse Ecosystem

The main purpose of *Eclipse Developer's Journal (EDJ)* is to educate and inform users of Eclipse and developers building plug-ins or Rich Client Platform (RCP) applications. With the help of great columnists, news you can use, and technical articles, *EDJ* is a great information source that you will be able to use to educate yourself on just about anything Eclipse related

Subscribe Today!

**FOR YOUR DIGITAL
PUBLICATION**

(AVAILABLE IN DIGITAL FORMAT ONLY)

6 Issues for \$19.99

Visit our site at www.eclipse.sys-con.com or
call 1-888-303-5282 and subscribe today!

OFFER SUBJECT TO CHANGE WITHOUT NOTICE



<http://eclipse.sys-con.com>

SYS-CON.COM

SYS-CON Media, the world's leading publisher of IT-technology magazines for developers, software architects, and e-commerce professionals, brings you the most comprehensive coverage of Eclipse

Compare this fairly straightforward code to the ExampleGateway.java, and you can see that more or less all we added is the pollForNewMessages(), the setting of various variables, and the creation of the event that passes NewMessageCount. All the gateway does is check the POP3 server every 60 seconds and, if there is mail flagged as new, sends an event to the associated listener CFC. Another thing you might notice is the usage of the Gateway logger class. Developing custom gateways can be difficult in that you often have to run the gateway on ColdFusion before you can tell if there are any problems. ColdFusion will not return much in the way of debugging information so it's a good idea to use the logger class to output information to aid in the development of your code.

Now that we have developed the code for our gateway, let's deploy this new gateway type via the ColdFusion Administrator.

Deploying a Custom Event Gateway

Deploying an event gateway is about

as easy as creating a new gateway instance. First you need to compile the POP3Gateway and place it in a .jar file, then make sure you place the .jar file in the cf_root\gateway\lib. For this example, you also need to make sure the J2EE.jar and the POP3Gateway.jar are in this directory.

Note: On J2EE configurations, you want to put your .jars in cf_root\WEB-INF\cfusion\gateway\.

When you're sure that your .jar files are in the right place, go to the ColdFusion Administrator and click on Event Gateway and then on Gateway Types.

The first field is the event Type Name. Enter POP3Gateway. The Description field should show a description of the event type in the Event Gateways > Gateway Instances creation screen, so for this example, use Test of Gateway using POP3. Then enter the .jar name in the Java Class field – POP3Gateway in this example. For Startup Timeout, leave the default of 30 seconds. Leave the Stop on Startup Timeout option checked.

Your finished page should look something like Figure 5.

Now click on Add Type to deploy your new event gateway type. You'll see the POP3Gateway event type now under Configured ColdFusion Gateway Types.

With the gateway deployed, you can test it by using Listing 5, creating a config file, and setting up a new event instance.

For this example, you can create a configuration file called pop3gateway.cfg and add your POP3

server's hostname, your e-mail login, and your e-mail password like this:

```
hostname=mail.robisen.com
login=robisen
password=gl0reibel2!
```

Save these files in an appropriate location, and then make a new event instance in the ColdFusion Administrator. When you test this gateway, you should add a new log file and record whether you have any new e-mails in your POP account. You could make this example more interesting by forwarding the message about the number of new e-mails to your IM or SMS account. To make the Java gateway example more interesting, let it provide more information or allow you to send e-mail or do anything else that POP3 allows you to do. Although CFMAIL and CFPOP offer a lot of functionality, creating your own POP gateway lets you add greater functionality and work with your POP accounts in asynchronously.

ColdFusion gateways are the most exciting and powerful feature to be added to ColdFusion since the original CF-QUERY tag. With ColdFusion gateways, you can connect to almost anything and develop a whole new type of ColdFusion application.



About the Authors

Ben Forta is Adobe's evangelist for the ColdFusion product line. He is the author of several books.

Sarge Sargent is a senior technical support engineer, Adobe Systems, based in Gilbert, Arizona. He has been coding advanced applications in CF since version 2.0.

ben@forta.com

ssargent@adobe.com

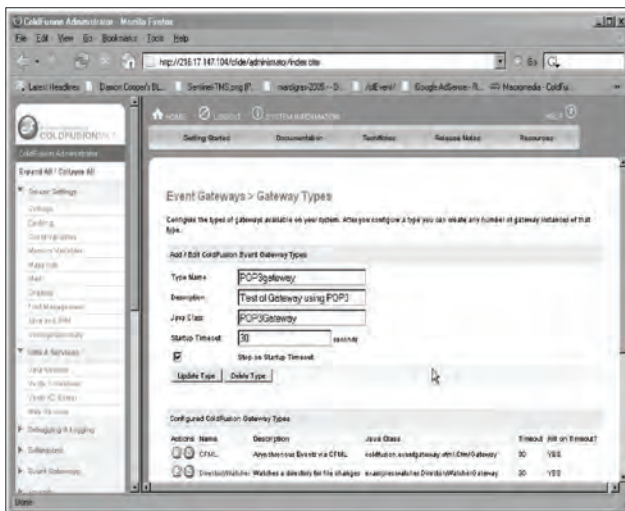


Figure 5: The POP3Gateway event type before it has been deployed in the ColdFusion Administrator.

Listing 1: DirectoryWatcher.cfc – Simple CFC that creates thumbnails of new images

```
<cfcomponent>
<cffunction name="onAdd" output="no">
<cfargument name="CFEvent" type="struct">
<!-- get event data -->
<cfset data=CFEvent.data>
<!-- watcher will ignore outgoing messages -->
<!-- Location of images -->
```

```
<cfset GalleryFolder = ExpandPath("gallery")>
<!-- Get list of images -->
<cftry>
<cfdirectory action="list" name="GetImages" directory="#GalleryFolder#"
filter="*.jpg">
<!-- Loop over images -->
<cfloop query="GetImages">
<!-- Proposed location of thumbnail -->
<cfset ThumbPath = ExpandPath("gallery/thumbs/#Name#")>
```

```

<!-- If the thumbnail does not exist -->
<cfif not FileExists(ThumbPath)>
<!-- Invoke our image-resizing function -->
<cfinvoke component="JavaInteg.imageio.resize.ImageManipulator"
method="createResizedImage" sourcePath="#GalleryFolder#/#Name#"
destPath="#ThumbPath#" newWidth="100">
<!-- log a message -->
<cflog file="watch" text="a file was #data.type# and the name was
#data.filename#">
</cfif>
</cfloop>
<cfcatch type="Any">
<cflog file="watch" text="A exception, #CFCATCH.TYPE#, was thrown in
DirectoryWatcher.CFC, the error message is #cfcatch.message#">
</cfcatch>
</cftry>
</cffunction>
</cfcomponent>

```

Listing 2: DataLogger.cfc – Simple CFC that logs incoming CFEvents

```

<cfcomponent>
<cffunction name="onIncomingMessage" output="no">
<cfargument name="CFEvent" type="struct" required="yes">
<cfif not IsDefined("CFEvent.Data.file")><cfset
CFEvent.Data.file="defaultEventLog"></cfif>
<cfif not IsDefined("CFEvent.Data.type")><cfset CFEvent.Data.
type="info"></cfif>
<cflog text="#CFEvent.Data.message#"
file="#CFEvent.Data.file#"
type="#CFEvent.Data.type#"
thread="yes"
date="yes"
time="yes"
application="yes">
</cffunction>
</cfcomponent>

```

Listing 3: DirectoryWatcher.cfc – Addition of asynchronous logging requests

```

<cfcomponent>
<cffunction name="onAdd" output="no">
<cfargument name="CFEvent" type="struct">
<!-- get event data -->
<cfset data=CFEvent.data>
<!-- watcher will ignore outgoing messages -->
<!-- Location of images -->
<cfset GalleryFolder = ExpandPath("gallery")>
<!-- Get list of images -->
<cftry>
<cfdirectory action="list" name="GetImages" directory="#GalleryFolde
r#"
filter="*.jpg">
<!-- Loop over images -->
<cfloop query="GetImages">
<!-- Proposed location of thumbnail -->
<cfset ThumbPath = ExpandPath("gallery/thumbs/#Name#")>
<!-- If the thumbnail does not exist -->
<cfif not FileExists(ThumbPath)>
<!-- Invoke our image-resizing function -->
<cfinvoke component="JavaInteg.imageio.resize.ImageManipulator"
method="createResizedImage" sourcePath="#GalleryFolder#/#Name#"
destPath="#ThumbPath#" newWidth="100">
</cfif>
<cfscript>
logMessage = structNew();
logMessage.message = "watch";
logMessage.message = "a file was #data.type# and the name was
#data.filename#";
</cfscript>
<cfset logAppInfo = SendGatewayMessage("DataLogger", logMessage)>
</cfloop>
<!-- log a message -->
<cfcatch type="Any">
<cfscript>
logMessage = structNew();

```

```

logMessage.message = "watch";
logMessage.message = "A exception, #CFCATCH.TYPE#, was thrown in
DirectoryWatcher.CFC, the error message is #cfcatch.message#";
</cfscript>
</cfcatch>
</cftry>
</cffunction>
</cfcomponent>

```

Listing 4: POP3Gateway.java – Checks for new e-mail and creates an event

```

import coldfusion.eventgateway.CFEvent;
import coldfusion.eventgateway.Gateway;
import coldfusion.eventgateway.GatewayServices;
import coldfusion.server.ServiceRuntimeException;
import coldfusion.eventgateway.Logger;
import java.io.File;
import java.io.InputStream;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Hashtable;
import java.util.Properties;
import javax.mail.*;
import javax.mail.internet.*;
public class POP3Gateway implements Gateway
{
private GatewayServices gatewayService = null;
private String gatewayID = "";
private String[] listeners = null;
private String config = null;
private Thread listenerThread = null;
private boolean shutdown = false;
private int status = STOPPED;
private String hostname = null;
private String username = null;
private String password = null;
private long pollingInterval = 60000;
private long listenerThreadWait = 10000;
private Logger logger = null;
public POP3Gateway(String gatewayID, String config)
{
this.gatewayID = gatewayID;
this.config = config;
this.gatewayService = GatewayServices.getGatewayServices();
this.logger = gatewayService.getLogger("pop3");
this.loadPropertiesFromFile();
this.logger.info("POP3Gateway(" + gatewayID + ", " + config +
").constructor: complete");
}
public String outgoingMessage(coldfusion.eventgateway.CFEvent cfmsg)
{
return "We have no outgoing messages from this gateway.";
}
public void setCFCListeners(String[] listeners)
{
this.listeners = listeners;
}
public coldfusion.eventgateway.GatewayHelper getHelper()
{
return null;
}
public void setGatewayID(String id)
{
gatewayID = id;
}
public String getGatewayID()
{
return gatewayID;
}
public void start()
{
this.logger.info("POP3Gateway.start():enter");
this.status = STARTING;
// Start up listener thread
Runnable r = new Runnable()
{

```

```

public void run()
{
    pollForNewMessages();
}
};
this.listenerThread = new Thread(r);
this.shutdown = false;
this.listenerThread.start();
this.status = RUNNING;
this.logger.info("POP3Gateway.start():exit");
}
public void stop()
{
    this.logger.info("POP3Gateway.stop():enter");
    this.status = STOPPING;
    this.shutdown = true;
    try
    {
        listenerThread.interrupt();
        listenerThread.join(this.listenerThreadWait);
    }
    catch (InterruptedException e)
    {
        // ignore
    }
    this.status = STOPPED;
    this.logger.info("POP3Gateway.stop():exit");
}
public void restart()
{
    stop();
    loadPropertiesFromFile();
    start();
}
public int getStatus()
{
    return status;
}
private void loadPropertiesFromFile() throws ServiceRuntimeException
{
    this.logger.info("POP3Gateway.loadPropertiesFromFile():enter");
    Properties properties = new Properties();
    try
    {
        FileInputStream propsFile = new FileInputStream(config);
        properties.load(propsFile);
        propsFile.close();
    }
    catch (IOException e)
    {
        String error = "POP3Gateway (" + gatewayID + ") Unable to load configuration file";
        throw new ServiceRuntimeException(error, e);
    }
    this.hostname = properties.getProperty("hostname");
    this.username = properties.getProperty("username");
    this.password = properties.getProperty("password");
    this.logger.info("POP3Gateway.loadPropertiesFromFile():exit");
}
private void pollForNewMessages()
{
    this.logger.info("POP3Gateway.pollForNewMessages():enter");
    int lastMessageCount = 0;
    Store store = null;
    Folder folder = null;
    try
    {
        Properties properties = new Properties();
        Session session = Session.getDefaultInstance(properties, null);
        store = session.getStore("pop3");
        store.connect(hostname, username, password);
    }
    catch (javax.mail.MessagingException e)
    {
        throw new ServiceRuntimeException(e.getMessage());
    }
    while (!shutdown)
    {
        this.logger.info("POP3Gateway.pollForNewMessages():testing for mail");

```

```

        int newMessageCount = 0;
        try
        {
            folder = store.getFolder("INBOX");
            folder.open(Folder.READ_ONLY);
            newMessageCount = folder.getMessageCount();
            folder.close(false);
        }
        catch (javax.mail.MessagingException e)
        {
            throw new ServiceRuntimeException(e.getMessage());
        }
        this.logger.info("POP3Gateway.pollForNewMessages():new message count=" + newMessageCount);
        if (lastMessageCount != newMessageCount)
        {
            this.logger.info("POP3Gateway.pollForNewMessages():lastMessageCount==" + lastMessageCount + "; newMessageCount==" + newMessageCount);
            this.sendMessageCountToCF(newMessageCount, lastMessageCount);
        }
        lastMessageCount = newMessageCount;
        try
        {
            Thread.sleep(this.pollingInterval);
        }
        catch (InterruptedException e)
        {
            // ignore
        }
        try
        {
            folder.close(false);
        }
        catch (Exception e)
        {
        }
        this.logger.info("POP3Gateway.pollForNewMessages():exit");
    }
    private void sendMessageCountToCF(int newMessageCount)
    {
        this.logger.info("POP3Gateway.sendMessageCountToCF(" + newMessageCount + "):enter");
        CFEvent cfEvent = new CFEvent(gatewayID);
        cfEvent.setCfcMethod("newMailCount");
        Hashtable returnedData = new Hashtable();
        returnedData.put("NEWMAILCOUNT", Integer.toString(newMessageCount));
        cfEvent.setData(returnedData);
        cfEvent.setGatewayType("POP3Gateway");
        cfEvent.setOriginatorID("POP3Gateway");
        // Send to each listener
        for (int i = 0; i < listeners.length; i++)
        {
            // Set CFC path
            cfEvent.setCfcPath(listeners[i]);
            // send it to the event service
            gatewayService.addEvent(cfEvent);
        }
        this.logger.info("POP3Gateway.sendMessageCountToCF(" + newMessageCount + "):exit");
    }
}

```

Listing 5: POP3Gateway.CFC – Simple CFC to log messages from the POP3gateway.

```

<cfcomponent>
<cffunction name="newMailCount" output="no">
<cfargument name="CFEvent" type="struct" required="yes">
<cfset data = CFEvent.data>
<!-- NEWMAILCOUNT -->
<cflog file="pop3gateway" text="you have #data.NewMailCount#">
</cffunction>
</cfcomponent>

```

Download the Code...
Go to www.coldfusionjournal.com

BALANCE

Designer/developer, front-end/back-end, clients/sanity. . .web development is a balance and we can help you maintain it. Join now and experience a wealth of training resources tailored to the tools you use every day.

www.communitymx.com



Visit www.communitymx.com/trial/ for your free 10 day trial.



Intermedia.NET...

Now serving the hottest ColdFusion.

© Copyright INTERMEDIA.NET, Inc 2005. All rights reserved. All other trademarks are property of their respective holders.

At Intermedia.NET we go beyond the industry standard by supporting the hottest new Coldfusion software, offering power like never before. For nearly a decade, we've been providing reliable, secure hosting to thousands of companies across the globe. We can do the same for you.

Intermedia.NET's premier hosting services include:

- ColdFusion MX hosting
- Custom tag registration
- Competitive plans
- Verity collections search engine
- Security sandboxes
- Guaranteed service levels

Unprecedented power, unmatched reputation...

Intermedia.NET is your hosting solution.



INTERMEDIA.NET

Call us at: 1.888.379.7729

e-mail us at: sales@intermedia.NET

Visit us at: www.intermedia.NET